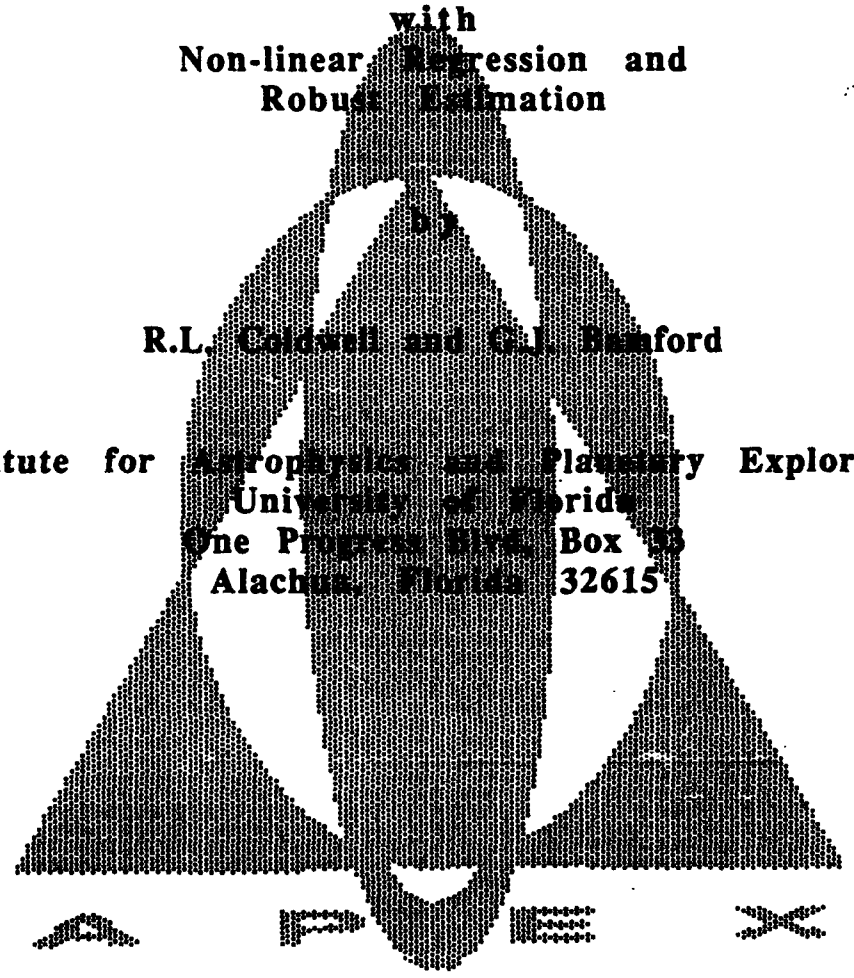
(1)

ROBFIT: A Code For the Analysis of Histogram Spectra with Non-linear Regression and Robust Estimation

by

R.L. Coldwell and G.J. Bamford

Institute for Astrophysics and Planetary Exploration
University of Florida
One Progress Blvd. Box 93
Alachua, Florida 32615



Final Report

Defense Advanced Research Projects Agency
Nuclear Monitoring Research Office
Under Grant No. N00014-87-J-1259
Monitored by Department of the Navy
Office of Naval Research

ROBFIT: A Code For the Analysis of Histogram Spectra

with

Non-linear Regression and

Robust Estimation

by

R.L. Coldwell and G.J. Bamford

Institute for Astrophysics and Planetary Exploration
University of Florida
One Progress Blvd, Box 33
Alachua, Florida 32615

Final Report

Defense Advanced Research Projects Agency
Nuclear Monitoring Research Office
Under Grant No. N00014-87-J-1259
Monitored by Department of the Navy
Office of Naval Research

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1a. REPORT SECURITY CLASSIFICATION: Unclassified	1b. RESTRICTIVE MARKINGS: N/A
2a. SECURITY CLASSIFICATION AUTHORITY: N/A	3. DISTRIBUTION/AVAILABILITY OF REPORT: Unlimited
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE: N/A	
4. PERFORMING ORGANIZATION REPORT NUMBER(S): N/A	5. MONITORING ORGANIZATION REPORT NUMBER(S): N/A
6a. NAME OF PERFORMING ORGANIZATION: University of Florida; 6b. OFFICE SYMBOL (If applicable):	7a. NAME OF MONITORING ORGANIZATION: The Office of Naval Research
6c. ADDRESS (City, State, and ZIP Code): Institute for Astrophysics and Planetary Exploration, One Progress Blvd, Box 33, Alachua, FL 32615	7b. ADDRESS (City, State, and ZIP Code): Space Physics Program, 800 N. Quincy Street, Arlington, VA 22217
8a. NAME OF FUNDING/SPONSORING ORGANIZATION: Defense Advanced Research Projects Agency; 8b. OFFICE SYMBOL (If applicable): NMRO	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER: Grant N00014-87-J-1259, R&T No. JD14004
8c. ADDRESS (City, State, and ZIP Code): 1400 Wilson Boulevard, Arlington, VA 22209-2308	10. SOURCE OF FUNDING NUMBERS: PROGRAM ELEMENT NO. N/A; PROJECT NO. N/A; TASK NO. N/A; WORK UNIT ACCESSION NO. N/A
11. TITLE (Include Security Classification): (U) ROBFIT, A Code For the Analysis of Histogram Spectra	
12. PERSONAL AUTHOR(S): R. L. Coldwell and G. J. Bamford	
13a. TYPE OF REPORT: Final Technical; 13b. TIME COVERED: FROM Jun 1 87 TO Aug 31 90; 14. DATE OF REPORT (Year, Month, Day): 05/04/90; 15. PAGE COUNT: N/A	
16 SUPPLEMENTARY NOTATION: N/A	

17. COSATI CODES		
FIELD	GROUP	SUB-GROUP

18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number): spectra analysis, non-linear minimization, robust estimation, least-squares fitting.

19 ABSTRACT (Continue on reverse if necessary and identify by block number)

This book contains the information needed to use the spectral fitting code ROBFIT. It explains the physical and mathematical foundations of the algorithms used in the code and describes to apply the code to real spectra. ROBFIT is a computer automated spectrum analysis package which tries to extract the maximum information from a histogram of intensity versus channel number. A non-linear minimization routine which uses a generalization of the Newton-Raphson technique has been developed and tested at the University of Florida's Department of Physics and Institute for Astrophysics and Planetary Exploration. This algorithm, simultaneous equation solver (SMSQ), is the central core of ROBFIT; SMSQ ensures the code finds the minimal fit, even with large numbers of non-linear constants in the fit. The intended user of the code is the person needing quick and accurate peak fitting with little user intervention. We have endeavored to make the code as "user tolerant" as possible and to this end we have included a user's guide in the book.

20 DISTRIBUTION/AVAILABILITY OF ABSTRACT: ☐ UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT. ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION: Unclassified
22a. NAME OF RESPONSIBLE INDIVIDUAL: R. Gracen Joiner	22b. TELEPHONE (Include Area Code): (202) 696-4203; 22c. OFFICE SYMBOL: ONR

DD Form 1473, JUN 86 Previous editions are obsolete. SECURITY CLASSIFICATION OF THIS PAGE

Preface

This book contains the information needed to use the spectral fitting code, ROBFIT. It explains the physical and mathematical foundations of the algorithms used in the code and describes how to apply the code to real spectra. ROBFIT is a computer automated spectrum analysis package which tries to extract the maximum information from a histogram of intensity versus channel number. A non-linear minimization routine which uses a generalization of the Newton-Raphson technique has been developed and tested at the University of Florida's Department of Physics and Institute for Astrophysics and Planetary Exploration. This algorithm, simultaneous equation solver (SMSQ), is the central core of ROBFIT; it ensures that the code finds the minimal fit even with large numbers of non-linear constants in the fit.

The intended user of the code is the person needing quick and accurate peak fitting with little user intervention. We have endeavored to make the code as "user tolerant" as possible. A users guide, listed as Appendix B, will enable the reader to start running the code immediately. Spectral analysis, however, is still not quite that simple. To extract the maximum detail from the data, a great deal of information must be provided. This includes information on peak types, how the peak widths of each type vary with channel number, and estimates of the deviations to be expected from these estimates for the various peak types. The code uses this information during the fitting process to produce a file of peak locations, widths and strengths; plus an estimate of the errors in each of these. Chapters 3-6 explain the physics involved in the input parameters used by the code. Chapter 7 explains the method used to generate the error estimates and the limits of their validity. Chapter 8 contains analyzed sample spectra which should enable the reader to see the practical limitations of the code.

Acknowledgments: This work was supported by the Defense Advanced Research Projects Agency, Nuclear Monitoring Research Office through grant number N00014-87-G-1259 monitored by the Office of Naval Research.

The initial development of the ROBFIT users guide and the development of an IBM main frame vectorized version of ROBFIT were supported by an IBM grant to the University of Florida and the Northeast Regional Data Center.

Contents

1 Introduction

This introductory chapter will help prospective users of the code quickly decide whether ROBFIT meets their needs.

The first section gives an overview of how the code operates, and the second section goes into this operation in more detail. Reading this chapter introduces the user to the basics of running ROBFIT. The following chapters contain information on specific areas of the code. Once you have become familiar with the general operating characteristics of the code, you can skip to the chapter that you are most interested in. However, newcomers to spectral analysis are advised to read all chapters.

1.1 What is ROBFIT?

ROBFIT is a computerized spectral analysis package. Spectra are histograms of intensity versus channel number which contain peaks superimposed on an underlying background function. ROBFIT has been developed to aid the analysis of spectra containing many peaks which reside on a complex background function. ROBFIT runs on several computer systems. At the present time we have versions that run on IBM and Macintosh personal computers, and VAX and IBM mainframe computers. The code is written in FORTRAN and is almost entirely self contained. The major changes needed to adapt it to new operating systems can be found in a separate file. This file also contains the graphics commands for viewing the fitted data.

ROBFIT performs a least-squares fit to spectral data using user-chosen peak shapes and a background function composed of cubic splines. Cubic splines can be thought of as piecewise third order polynomials. During analysis, the entire background is fitted to a single spline function. This technique allows the background fit to follow complex fluctuations in the data and to use all possible information in the fit. This results in a more accurate representation of the background. We provide an introduction to this cubic spline fitting in Chapter 3. An algorithm called the simultaneous equation solver (SMSQ) has been written to minimize, in a least-squares sense, the difference between the data and the cubic splines of the fit to the data. Full details of this minimization procedure are given in Chapter 4. SMSQ

is an enhanced minimization scheme which means that ROBFIT can analyze spectra even with many constants in the fit. Quick fits to look at the gross properties of a spectrum can be completed in minutes on a personal computer. The code is configured so that it can run for long periods of time, succesively analyzing spectrum 1, spectrum 2, etc. This is necessary because on smaller personal computer systems a fit to the noise level of 100 peaks and a background composed of twenty constants, needs three to four hours to complete. In fact, we have used the code to analyze spectra which called for as many as 100 hours. After this period, the user has a representation of the data that is statistically correct and a fit that has used all the available data in its peak and background determinations. Many spectral analysis packages require the spectrum to be broken into small sections, each of which is then fitted separately. Since these packages need substantial user intervention, analyzing a 100 peak spectrum would require much more time than that taken by ROBFIT. Also, because these analyses are done on a small scale, the resulting fits may misrepresent the background continuum in regions with multiple peaks or complex background functions. A further advantage of ROBFIT is that the code can represent any peak shape. Common peak shapes such as Gaussian, Lorentzian or Voigt profiles are supplied with the code, but the user can generate other shapes by either picking a clean peak from the data and generating a peak shape from that, or by fitting some previous calibration of the peak shape. Peak shapes are built from back-to-back cubic splines which the code then uses as a representation of that peak shape during the fitting phase. A maximum of five independent peak shapes can be used in a single fit.

1.2 A more detailed view of ROBFIT operation

In its operation, ROBFIT separates spectra into two functions; background and foreground. The background contains slowly varying features of the spectra while the foreground contains the high frequency content. Accurate separation of these functions allows the code to detect small peaks and decompose multiple peak structures. ROBFIT iterates on background and foreground fitting to move smaller peaks from the background to the foreground.

The background is fitted over the entire spectrum as a set of cubic splines with adjustable knots. A knot is the place at which two cubic splines

meet. This background fitting is explained further in Chapter 5. Fitting over the whole spectral range allows the background features to be continuously fitted with fewer constants, resulting in a more accurate representation than is possible when they are fitted in small sections with the peaks. There are two algorithms that make this possible. The first, data compression, uses a robust averaging technique to reduce contributions to the background from peaks and spurious high points. The second, the minimization algorithm SMSQ, minimizes chi-square (χ^2) with respect to the constants of the background and foreground. Having represented the background as a smoothly varying function, peaks can be identified as regions of the spectra which lie above this background curve.

The foreground is the peak content of the spectra which may contain many peaks of various shapes. Each shape is represented by a collection of back-to-back cubic splines. These shapes are then fitted to the spectrum to determine where the peaks reside. Overlapping peaks, of various shapes, are fitted simultaneously. This process is explained further in Chapter 6. A fast residual locater for finding peak regions, a routine for systematically controlling the allowed widths, and SMSQ are the principle algorithms used in peak determination.

To control ROBFIT, the user chooses the number of constants to be fitted to the background and picks the peak shapes to be used. A typical first run of the code, illustrated in Figure 1, could proceed as follows:

a) Select a peak shape. A Gaussian could be chosen for the first run of the code. Though this may not be the correct shape, the code will still operate. The output can be checked for an incorrect peak shape, once fitting has ended.

b) Run the main code using the default options and the shape just generated. A large CUTOFF can be chosen for a first run. This has the effect of identifying only large peaks within the spectrum.

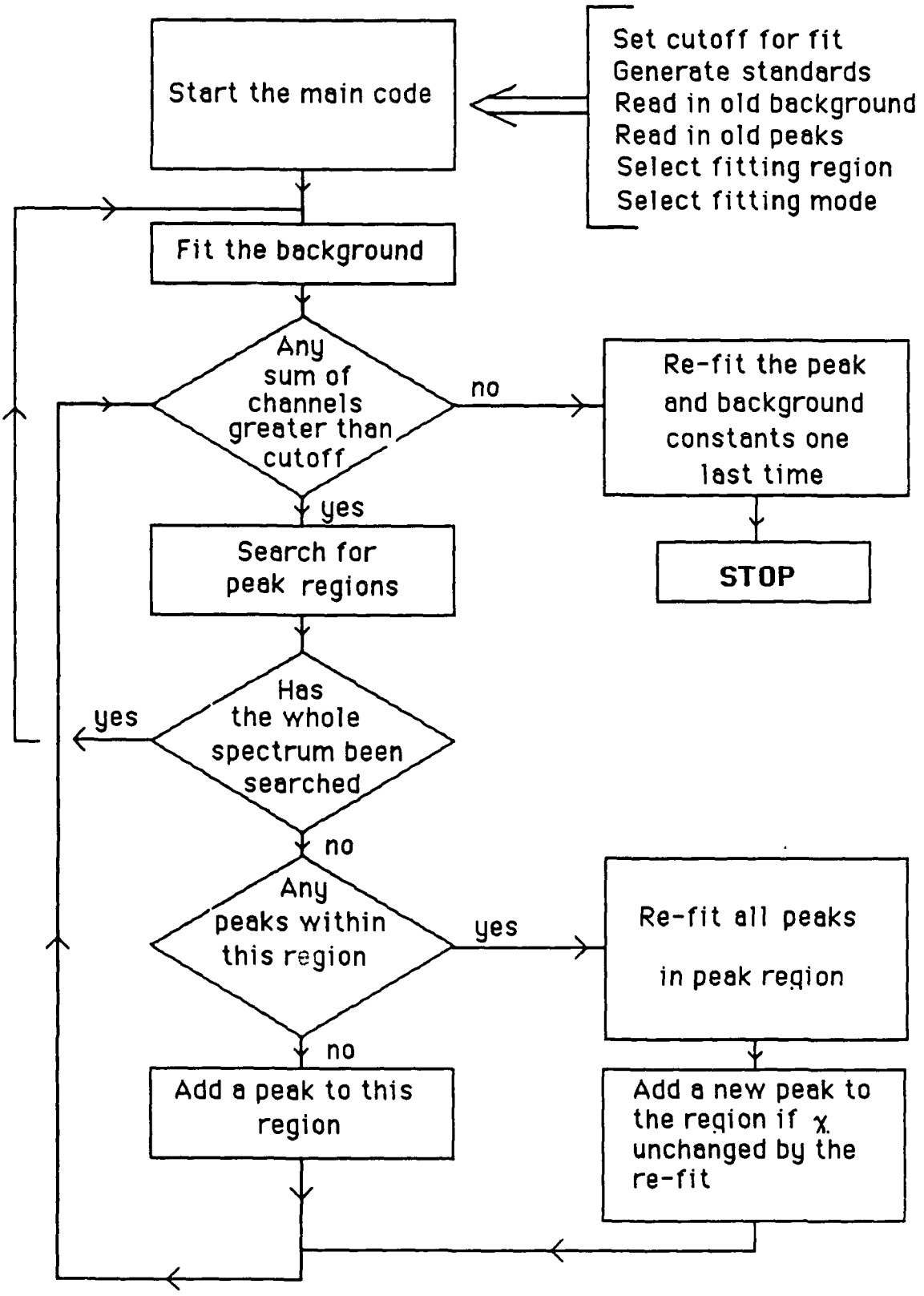


Figure 1 The general flow of ROBFIT

ROBFIT will fit the data and generate peak and background constant files and a graphical file for viewing the fit. Several cycles of fitting different peak shapes with various CUTOFF's may be needed to find the correct peak shapes in the data. During this cycling the user can also monitor the background function. By increasing the number of background constants in the fit, an optimum number can be decided upon. Once these preliminary quantities have been set, the CUTOFF can be lowered to enable detection of smaller peaks.

The code works well when detecting small peaks within a spectrum, where small means the same size as the local background fluctuations. It also performs well in regions with low or complex backgrounds. Under these conditions, the effects of imperfections in the representation of the peaks are negligible. Problems can arise when the data contains large peaks on a smooth background. Here the effects of peak shape variations may disrupt the fitting. Although the code can adjust for peak height and width variations, it cannot change the shape of the peak during fitting. It assumes the peak shape stays the same across the entire spectral range. Any variation in this peak shape will lead to a bad fit. A bad fit to a large peak has a significant effect on the χ^2 minimization and will limit the codes ability to home in on the best fit. Generally, this can be quite readily seen as the code breaks the large peaks into many smaller ones; a situation which we call braiding. When this happens the user must examine closely the form of the peak shape being fitted. Shape variation can be accounted for, to some extent, by generating several peak shapes for various channel regions. Now, as the code scans through the spectrum, it can choose a peak shape that is closer to the actual data peak shapes.

Throughout the analysis, ROBFIT keeps track of the errors introduced into each of the constants. This ensures that peak positions, widths, and strengths are statistically correct when the program ends. This error treatment is explained further in Chapter 7.

Though the code has been mainly used in the study of gamma-ray spectra, we expect that it will be of use in many areas of spectral analysis. In Chapter 8 we highlight the flexibility of the code by fitting a set of data samples taken from computer generated test cases, gamma-ray astrophysics and solar seismology.

The above has given a brief glimpse of the computer techniques used

by ROBFIT when analyzing a spectrum. To introduce the reader to some popular methods used in other spectral analysis codes, we give a brief review of computerized spectral analysis in Chapter 2. In Chapter 9 we discuss when we would consider using other analysis packages.

Appendix A contains a full listing of the code for those users who wish to study the actual coding of the algorithms used by ROBFIT. This listing will be used by users who wish to pirate certain parts of the code. The minimization routine SMSQ, for example, can be used outside ROBFIT as a general minimization algorithm. An illustration of this is given in Appendix C. For those readers wishing to get their hands dirty straight away, Appendix B contains a users-guide to ROBFIT. Appendix C contains a listing of a non-linear least-squares fitting routine that uses SMSQ. We call this routine NLFIT, and it is described in Chapter 4.

2 Review of spectral fitting

This chapter gives an overview of computer methods used in spectral analysis. We concentrate on gamma-ray spectra throughout this book. This is partly because ROBFIT was developed on these data and partly because, by their very nature, they provide stringent tests for any analysis package. However, we hope the code can be used in other areas of spectral analysis.

We begin by discussing the requirements of any spectral analysis code, and show how these requirements are fulfilled in the more popular analysis packages. Finally, we illustrate how ROBFIT performs these standard procedures.

2.1 A typical spectrum

ROBFIT has been used as a data reduction package in the Antarctic research program to study gamma-rays produced by Supernova 1987A[1]. Figure 2.1 shows a typical gamma-ray spectrum from the supernova illustrating the complexity of these spectra. Some regions contain well separated peaks while others have overlapping peaks.

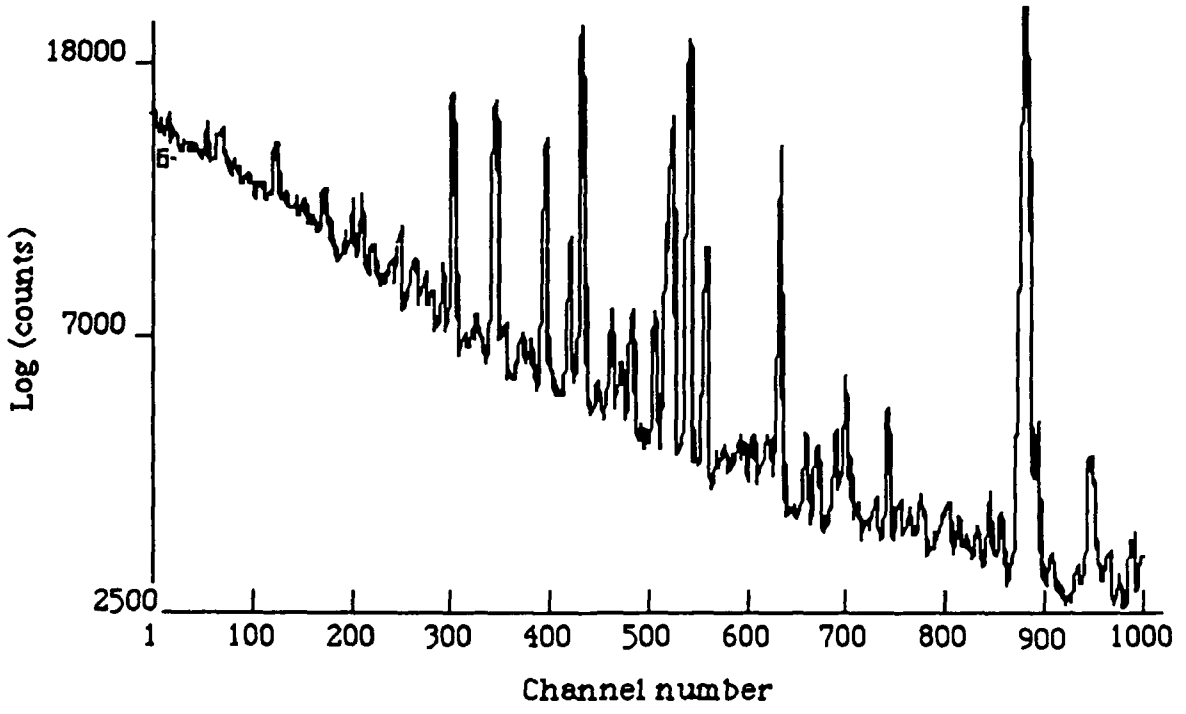


Figure 2.1 A typical gamma-ray spectrum from Supernova 1987A.

The interesting features in these spectra are not the easily identified large peaks, but the small ones the same size as the noise. In the supernova case, the signals also fell in regions of overlapping peaks. Figure 2.2 shows an actual supernova signal identified by ROBFIT.

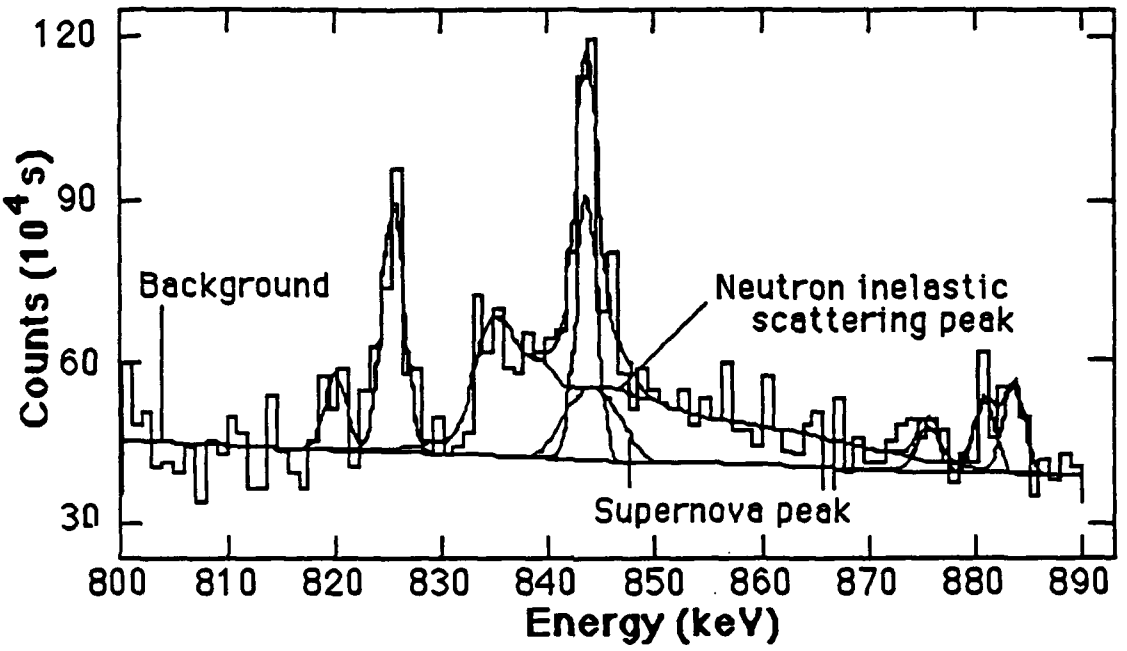


Figure 2.2 View of a small channel region surrounding the supernova signal.

Refer to Chapter 8 for the analysis of the supernova data. We mention this analysis here only to illustrate the level to which the fitting must be taken to extract these very small signals.

The goals of analysis of such spectra can be summarized as to:

a) correctly determine all single peak parameters,
b) correctly separate peaks in overlapping peak regions, and
c) provide accurate analysis in the presence of noise.

The reader interested in learning more about gamma-ray spectra, is referred to a book by Knoll[2] which gives a comprehensive review of the subject.

Though the supernova data are complicated, a limited analysis can be done manually by either graphically viewing the data and fitting functions by eye, or by applying stripping techniques[3]. Stripping techniques are used to successively subtract individual components from the spectrum. These

methods, however, are laborious and prone to error. A computer analysis provides a more accurate evaluation and also automates the peak finding process, thus easing the burden on the user. The following section gives a brief review of computer fitting techniques.

2.2 Computerized fitting

The aim in fitting any spectrum is to identify "all" the peaks. Before any peaks can be found, the underlying background must be estimated. Separating the peak and background functions, however, is a difficult task. When fitting a peak, the background must be accurately determined, but in determining the background, all the peaks need to be identified. The various analysis routines break this cycle in many ways. In this section we do not intend to review each of these routines in detail, but rather to give an overview of the techniques used. We will concentrate only on the areas of peak and background identification.

Background determination is generally done in one of two ways:

a) By considering only small regions of the spectrum, and assuming the background is a simple function that can be fitted with a low order polynomial.

b) By taking the whole spectrum, and in some way filtering out the peaks to leave a smooth background function.

The first method has been used in the earliest computerized analysis programs. Its simple low order polynomial fit makes it easy to encode. However, there are dangers with such an approach. The true background may call for more than just the low order polynomial to describe it. This is especially so when the region contains multiple peaks or a rapidly changing background function. This means that the background will interfere with the determination of peak parameters, such as widths, areas and position. Even so, the more popular spectral analysis codes[4,5,6] still use this procedure.

The second method is the more desirable approach to background fitting. Here all the spectral information is available, supplying a better representation of the background. Several techniques have been used to perform this automated background calculation. The two most popular methods are listed here.

a) Averaging techniques are used to smooth the background continuum. Points that lie outside a specified number of standard deviations from the smoothed data are replaced by the smoothed value. This averaging effectively filters out the high frequency content of the peaks. Iterating on this procedure progressively removes peaks and leaves a smoothed background function.

b) Minimal search techniques use the fact that peak free regions of a spectrum contain less counts than neighboring peak regions. A search for background minima can be carried out by determining "local minima" over the entire spectral range on a channel by channel basis. A local minima is identified as any channel for which the number of counts in that channel are less than adjacent channels. The local minima are then connected using an interpolation scheme to give the background function.

A review of four of the more widely used background estimating codes is given in reference 7.

Once the background has been calculated, a search for peaks can begin. Unlike the background, the peak shapes may be known *a priori*, or if not, then approximated by some standard function. It is generally enough to simply find the peak positions, and then insert the peaks at those points. A least-squares fit can then be used to better determine peak widths, strengths, and positions. The critical task is the accurate determination of peak positions. These are generally chosen by one of the following methods.

a) The first method looks for regions of high curvature within the data. Peaks can be detected by either:

1) studying the "first differences" which are calculated by subtracting a given channel value from the next highest channel value. Peaks appear here as a significant change in the sign of the difference. Or,

2) studying the "second differences" by measuring the degree of curvature at each channel position. Sharp photo peaks have high curvature while background, such as Compton continua, generally shows only mild curvature[8].

b) The second method performs a correlation between the data, and a given peak shape[9]. For efficient application, the width of the correlation function must be about the same as the width of a peak.

Whichever technique is used, the idea is to detect as many true peaks and as few spurious peaks as possible. Once a peak position has been found the next stage is to accurately determine all peak positions (peak centroids), widths and areas (number of counts under the peak). Most peak fitting procedures use as their basis, an iterative non-linear least-squares method for such determinations. Here the parameters that describe a peak shape are varied while fitting the shape to the data. The optimum values are determined by minimizing the χ^2 for the fit. These optimum values then describe the peak[10]. An introduction to least-squares fitting and the χ^2 function can be found in a book by Bevington[11].

The above peak fitting methodology has troubles if the actual peak shape is unknown. Generally, gamma-ray peak shapes can be approximated by a Guassian function, which is usually combined with some additional tailing criterion. However, other shapes are not so easily parameterized. One advantage to using ROBFIT is its ability to use the experimental data itself to accurately deal with odd peak shapes.

2.3 An even more detailed view of ROBFIT operation

ROBFIT contains six modes of operation; three for curve fitting and three for display of the data and curve fits.

For fitting we have a Code

a) for generating peak shapes for use in fitting,
b) to fit the background alone, and
c) to perform a full spectral fit.

For graphically viewing the fits we have Codes to

a) display the raw data,
b) display the peaks shapes, and
c) display the full spectral fit.

Earlier versions of the code have been described in two papers by Coldwell[12,13].

In Chapter 1, we gave a brief introduction to ROBFIT operation. We now give an even more detailed account to illustrate the code's background and peak fitting algorithms.

Determination of the background is carried out by smoothing the spectra. This involves compressing the data by replacing every sixteen channels by the average over that region. The average is a robust average which helps reduce remaining peak contributions. This smoothing has two functions, a) it filters out large data fluctuations, and b) because it reduces the number of points in the fit, it speeds up the background fitting. After smoothing, the entire spectral channel range is used to calculate the shape of the background. The user then chooses the number of constants to be fitted to this smoothed background. ROBFIT adds splines to its representation of the background, refitting after each spline until the user-supplied number has been reached. This background calculation can be interspersed with peak fitting when performing a full spectral fit. One must always be cautious and not over fit the background by specifying too many constants as this will allow it too much flexibility. In practice, the background can be kept "stiff," and constants added slowly to "slacken" it as needed. That way the background fit will not rise into the peak regions of the spectrum.

In its peak-fitting phase ROBFIT needs knowledge of the peak shapes it expects to find within the data. These shapes are termed standards. ROBFIT's standard-generating mode allows the user to either create a standard from the raw data or to use one of the three most common shapes, Lorentzian, Gaussian or Voigt. Each standard is fitted to a polynomial plus a set of back-to-back cubic splines. The polynomial is optional. It is used to represent the background under a peak when generating a standard from real data. The splines represent the standard peak. Complex-shaped standards can be generated using this technique.

ROBFIT uses the standards to search for peaks within the data. Each peak is determined by minimizing a weighted sum of the differences between the data and a background-plus-peaks function; with respect to the peak height, location, and width. After initial peak fitting has been carried out, the weights and the background-plus-peaks function are re-determined. The old peaks are refitted and new peaks are added. This cycle is repeated until there are no peaks greater than a user-specified level. Before finishing, all the fitted parameters are re-optimized to ensure correct representation of peaks and background. This iterative procedure has been outlined in Figure 1.

The full spectral-fitting routines form the core of the fitting code. They perform a complete spectral analysis on the data. After cycling the code, the peak positions, widths, and strengths are fully determined. ROBFIT is configured so that it can start with or without information on the background and/or peak parameters. For example, if the background has been determined in a previous run, this information can be entered at the beginning of the next run. In doing so the code will not have to re-determine the background coefficients, and the fit will be speeded up. Alternatively, the positions or widths of certain peaks may be known beforehand. In such a case, this information can also be linked into the program, again resulting in speeding up the fit, and possibly increasing the accuracy of the final peak parameters.

In summarizing the main features of ROBFIT we can say that:

a) any peak shape can be accommodated.

b) by using the entire spectrum to determine the background, a better background representation is obtained.

c) iterating on peak and background fitting mean that background contamination of peaks and vice versa does not become a problem.

d) re-optimizing all the splines in the fit after fitting gives a more accurate fit to the background and foreground functions and provides an accurate measure of the errors on these quantities.

e) the above means that extremely small peaks can be picked out of a noisy spectrum.

3 Why use cubic splines?

The key to ROBFIT's ability to decompose spectral features lies in its use of spline functions. The idea is that all background and foreground features can be represented by smooth functions. The splines are the way in which these smooth functions are created. In this chapter we review the properties of cubic splines, and illustrate how they can be used to build up the spectral features. We start with a look at polynomial fitting as an introduction to the techniques used in least-squares fitting. This is also used to illustrate when polynomial fitting breaks down. We then show how the use of cubic spline functions can eliminate some of these problems.

3.1 Why not use a simple polynomial?

The usual way one fits an unknown function of data points y_i is to start by approximating the function by a polynomial. So that

$$y_{ap}(x) = c_0 + c_1x + c_2x^2 + \ldots\ldots\ldots\ldots c_Mx^M$$

This approximation is then fitted to the data using a least-squares fit. In a least-squares fit, one tries to minimize the difference between the data and the approximating function by choosing the optimum c_n constants. This is done by minimizing a set of differences

$$\Delta_{diff} = (y_i - y_{ap}(x_i))^2$$

where y_i and x_i represent one data point. An illustration of polynomial fitting is given in Figure 3.1. We go into the problems involved in polynomial fitting in a little more detail in Chapter 4.

The optimum set of constants can be calculated by minimizing the sum of the differences from each of the data points

$$Optm = Min\left(\sum \Delta_{diff}\right)$$

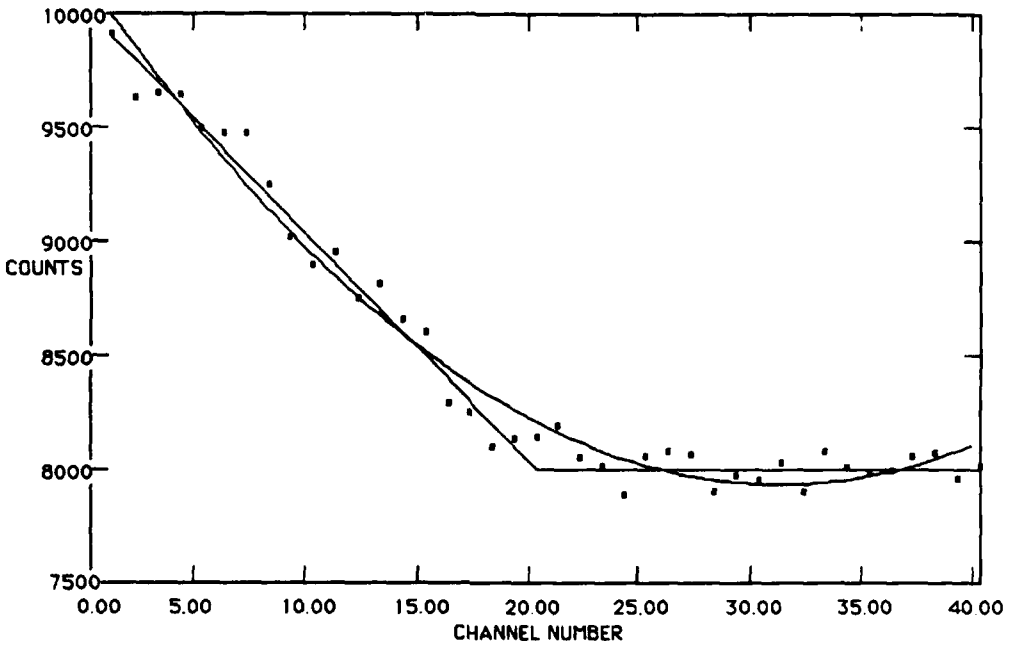


Figure 3.1 Illustration of polynomial fitting. The straight lines represent the function being fitted. The data points are randomly distributed about this function in a Gaussian manner. The curved polynomial fit is from a 4 constant fit to the data points.

To take the illustration of polynomial fitting a stage further, let us consider a specific example of a two constant (first order) polynomial fit. Here

$$y_{ap} = c_0 + c_1 x$$

or in the more usual format

$$y_{ap} = a + bx$$

which is a straight line fit to the data. We can now show some general features of polynomial fitting. Our minimizing function, χ^2 is

$$\chi^2 = \sum (y_i - a - b x_i)^2$$

which must be minimized with respect to the constants a and b. We can find the minimum for each of the constants by differentiating the minimizing

function with respect to that constant. The minimum for a is given by

$$\frac{\partial \chi^2}{\partial a} = -2\sum (y_i - a - bx_i) = 0$$

and for b it is

$$\frac{\partial \chi^2}{\partial b} = -2\sum \left\{ x_i((y_i - a - bx_i)) \right\} = 0$$

This gives us two simultaneous equations to solve. In the general case of a m^{th} coefficient fit, we end up with m simultaneous equations. This means that to fit the m coefficients, we need at least m data points. In our straight line fit this is clearly illustrated by the fact that we cannot fit a straight line to one point, but we can to two points. Using more than two points allows us to include additional information into our minimizing function, giving a better measurement of the constants. The simultaneous equations can be expressed in matrix form and solved (for the constants) by inverting a matrix. This matrix inversion can cause problems when the value of m is large.

When considering smooth curves for m values approximately five, or for m much less than the number of data points, the above minimization scheme works well. However, for large m the matrix becomes nearly singular, calling for extremely high precision in the inversion. Although these problems can be overcome by using orthonormal polynomials, a basic problem associated with high m fitting still persists. That is the tendency for the fit to loop though the data points as it over-fits the data. This "looping" is illustrated in Figure 3.2.

This phenomena is a numerical representation of the familiar Gibbs phenomena seen in looking at oscilloscope patterns of sharp waveforms. It is caused by the fact that in the fitting function all derivatives are continuous. In Figure 3.2 the large negative value of the first derivative before channel 20 continues for some distance past channel 20, and the resulting miss of the data is corrected by a positive first derivative to the right of channel 25, and so on. Normally, the problem occurs in derivatives higher than the first which we used for illustration purposes. The solution is to replace x^k by a spline of order L which has discontinuities in the L^{th} derivative.

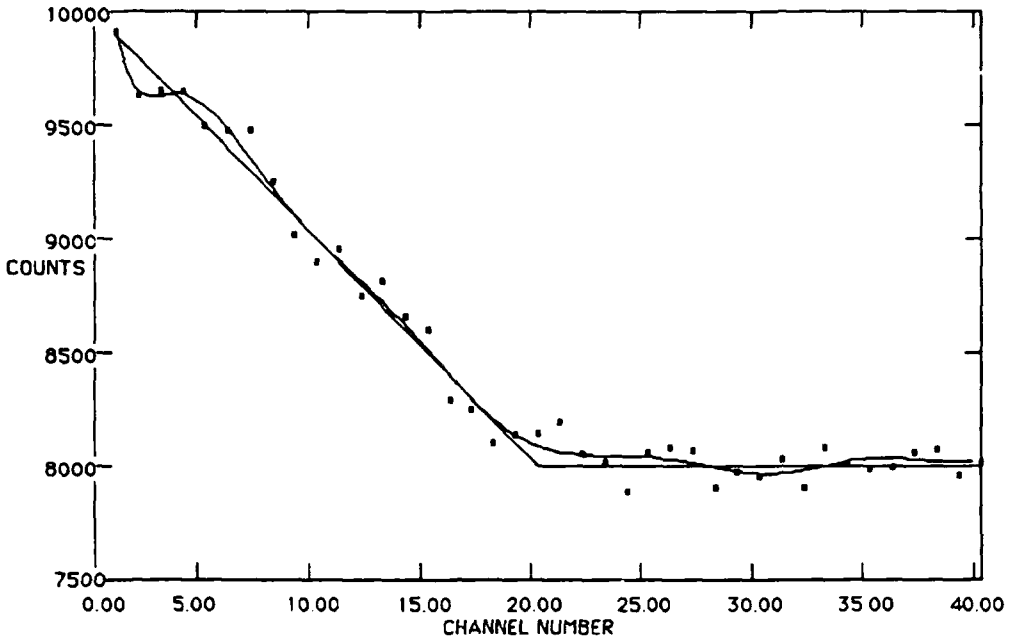


Figure 3.2 Illustration of "looping" caused by using too many constants in the fit.

3.2 Cubic splines and their properties

The spline function has been likened by Schoenberg[14] to the draftsman's spline, which is a flexible strip used for drawing smooth curves. In a mathematical spline, the strip is replaced by a sequence of polynomial functions which connect at points called knots. Any curve can then be portrayed by a sequence of these low order polynomials. The advantage of using a sequence of low-order polynomials is that complicated non-linear functions can be described, even in instances where they cannot be represented by a simple polynomial. In a cubic spline, the polynomial segments are of degree three. At the knots, cubic splines have the property that they are continuous in themselves and their first and second derivatives.

In our usage, a spectrum consists of a sequence of abscissa, ordinates, and standard deviations

$$\{ x_i \},\{ y_i \},\{ e_i \} \quad (i = 1, \ldots, N)$$

We assume this spectrum measures a series of peaks superimposed on a background. The object is to find properties of the peaks such as their location,

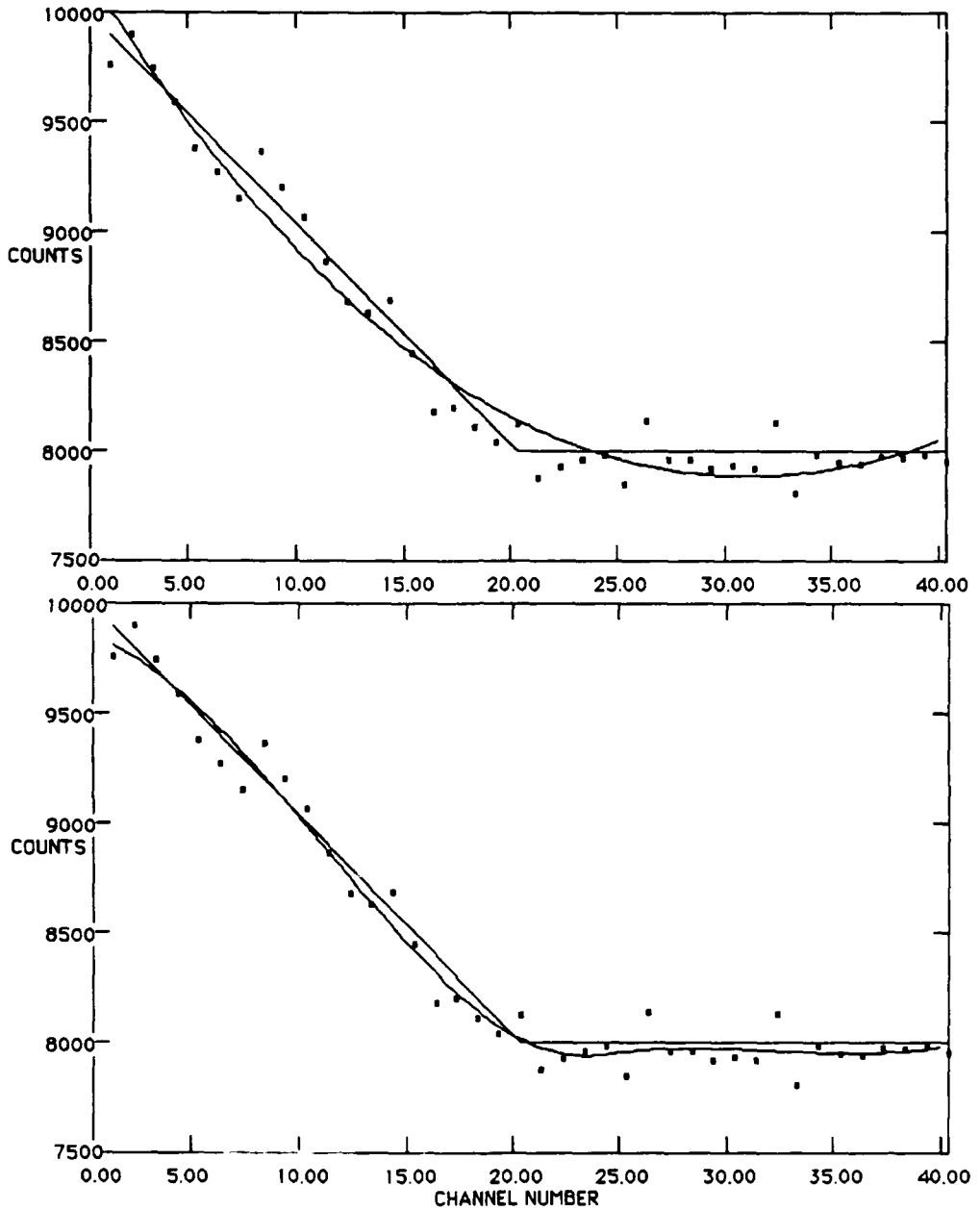


Figure 3.3 Effect of increasing the number of knots in a fit. The upper figure shows a 2 knot fit while the lower figure shows a 4 knot fit to the same data.

area, and width, the form of the background, and to determine the uncertainties in these properties. We use cubic splines to represent the peak and background features.

For example, the cubic spline used to represent the background can be written as

$$S(x) = \sum_{i=0}^{3} c_i x^i + \sum_{j=1}^{M} d_j (k_j - x)_+^3$$

with

$$(x)_+ = 0 \text{ for } x \leq 0$$
$$(x)_+ = x \text{ for } x > 0$$

where $\Sigma\, c_i x^i$ describes a third order polynomial background which is used as the basis for background fitting. The number of splines (M) to be included is set by the user. These add a contribution $\Sigma\, d_j (k_j - x)_+^3$ to the background representation. The c_i and d_j are the scaling constants for the polynomial and individual spline coefficients. The k_j's are the knot positions. Figure 3.3 illustrates the effect of increasing the number of knots in a particular fit.

The background function derivatives are

$$S'(x) = \sum_{i=1}^{3} c_i i x^{i-1} - 3 \sum_{j=1}^{M} d_j (k_j - x)_+^2$$

$$S''(x) = \sum_{i=2}^{3} c_i i(i-1) x^{i-2} + 6 \sum_{j=1}^{M} d_j (k_j - x)_+$$

$$S'''(x) = 6C_3 + 6 \sum_{j=1}^{M} d_j \theta(k_j - x)_+$$

where

$$\theta(x)_+ = 1 \text{ for } x > 0$$
$$\theta(x)_+ = 0 \text{ for } x < 0$$

These are plotted for M = 4, C = { 1, -1, 1, -1 } , d = { 1, -1, 1, -1 }

k = { .2, .4, .6, .8, }

in Figure 3.4. Note the fact that straight lines connect the points

S" (0, .2, .4, .6, .8,1.0) = (-0.4,-1.6,-1.6,-2.8,-2.8,-4.0)

The theorem of interest to us was proved by Holladay[15] in 1957 and is restated by Ahlberg, Nilson and Walsh[16] as Theorem (Holladay). Let $\Delta : a = x_0 < x_1 < \ldots < x_n = b$ and a set of real numbers $\{ y_i \}$ (i = 0, 1, ... N) be given. Then of all functions f(x) having a continuous second derivative on [a, b] and such that $f(x_i) = y_i$ (i = 0, 1, ... N), the spline function $S_\Delta (f_j\ x)$ with junction points at the x_i and with S_Δ " $(f_j\ a) = S_\Delta$ " $(f_j\ b) = 0$ minimizes the integral

$$\int_a^b |f''(x)|^2 \, dx$$

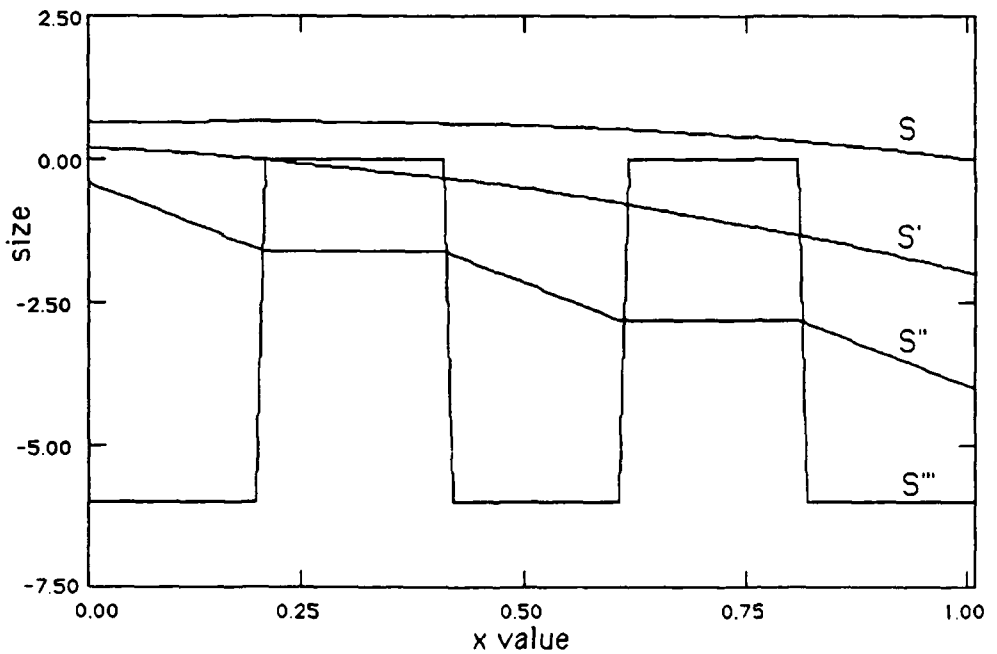


Figure 3.4 The background function S and its first, second and third derivatives.

In other words, the spline function is the function representing the data with the smoothest possible second derivatives. In essence, this is true because for splines, straight lines connect the relevant data points. These must be connected, but no others need be, so the straight lines minimize the integral.

Cubic splines have also been shown to converge in the interval (a, b) to all functions with continuous fourth derivatives such that

$$\| f\text{-}S \|_\infty \leq \frac{5}{384} \| f^{(4)} \|_\infty h^4$$

where f is the function, and S is the cubic spline

$$\| f^{(4)} \|_\infty$$

is the largest value of the absolute value of the fourth derivative in (a, b) and h is the maximum spacing between knots. Also, they converge to the derivatives $f^{(n)}$ with larger constants and lower powers h^{4-n}. This means that in any interval, enough knots guarantee convergence to the background. That is, if the data can be represented by an underlying function (in the mathematical sense) with continuous and defined fourth derivative, then it can be fitted to within our statistical error limits with a finite number of terms. This may not seem too valuable, but when all else is failing it can be somewhat comforting, and is frequently all that one has.

The cubic splines used in this book differ quantitatively from the splines discussed above. About 4000 data points will be fitted with splines containing between 10 and 50 knots. The smoothness of the fit will allow us to pick out sharp features which will normally be fitted separately as peaks. The desire to find the smallest possible peaks led to de Boor's statements on varying knots[17].

> "Use of a code for finding a (locally) best approximation from $S_{k,n}$ [the knot placement] is expensive. It is warranted only when precise placement of some knots is essential for the quality of the approximation (e.g., when the data exhibits some discontinuity) and an approximation with as few parameters as possible is wanted. Otherwise, an approximation with two or three times as many well-chosen knots is much cheaper to obtain and usually, just as effective."

The first of de Boor's conditions is true. Finding each knot location takes ROBFIT about fifty times as long as finding the constant associated with the knot. The background, however, is a cubic polynomial between knots and, therefore, subject to the Gibbs phenomena of oscillating through the data points shown in Figure 3.2. Extra knots imply regions in which there are constants not needed to fit the background.

To illustrate this point, in Figure 3.5 a single knot at 18 can be found by minimizing the χ^2 to about 36 with respect to the 6 constants in the fit. These constants include the location of the knot. A χ^2 of 45 can be found much faster by simply placing the knot at 24 as would seem to be indicated by the data. The fit responds by trying to keep the first derivative constant at 24 which artificially drives the curve low at 26, making the randomly high fluctuation at this value appear to be a spurious peak. To obtain our goal of maximum information, each constant in the background must be so involved in reproducing the structure of the background that it can not cause trouble. Thus, it is usually essential that the background be fitted with the minimum number of splines. Extracting the maximum information in this manner consumes more computer CPU time than would otherwise be called for to fit the

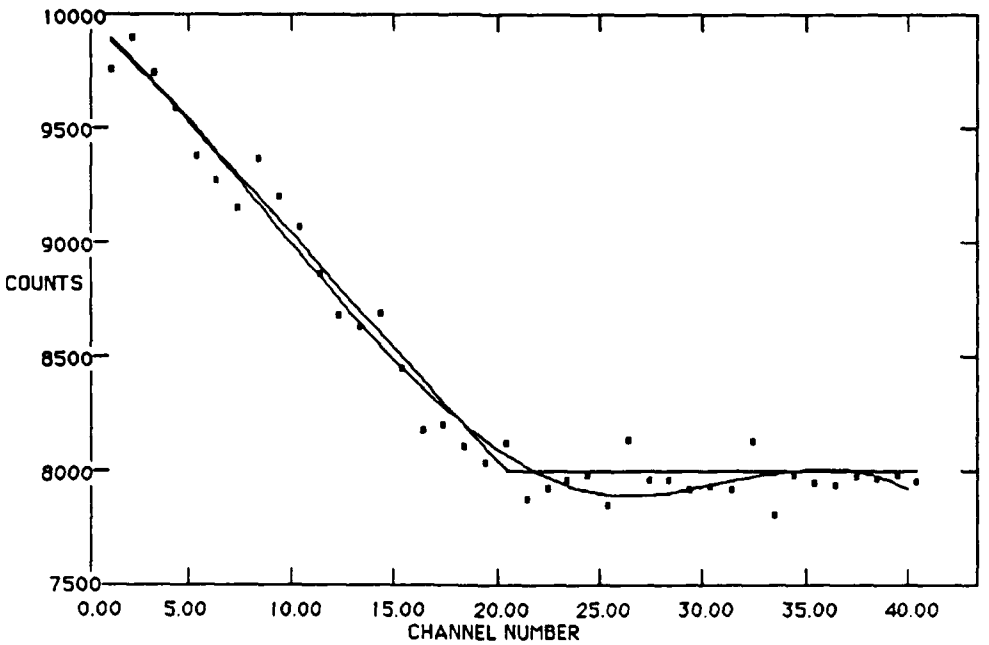


Figure 3.5 Spline fitting using a 4 constant polynomial plus single knot fit.

data. Once the knot locations have been determined, one can save time and gain accuracy, in some cases, by re-using them. This will be discussed further in Chapters 6 and 8.

One last comment about the splines used to fit the background is that frequently the background is positive definite. When this is the case, it can easily be incorporated by making the fitting function

$$f_A(x) = EXP\ (S(x)\ )$$

which cannot become negative. While retaining the local properties of S(x), $f_A(x)$ can vary by many orders of magnitude, and is therefore a better approximation to positive definite data. The *'LOG'* option which gives this in ROBFIT should be used unless the desired fit should have negative regions.

4 The non-linear minimization scheme

This chapter gives a review of weighted least-squares fitting and its applicability to spectral fitting. We introduce various methods for calculating channel weights and detail the minimization scheme used within ROBFIT. The final section illustrates how ROBFIT uses non-linear Newton-Raphson minimization and accelerated extrapolation routines to find the least-squares minimum.

4.1 A review of weighted least-squares fitting

The basic approach to any type of curve fitting is to design a figure of merit that measures the agreement between the data and a chosen model. This value is usually designed to become smaller as the agreement between data and model gets better. The model parameters can then be adjusted, to minimize the figure of merit, until optimum agreement between data and model is achieved. This is the basis of the least-squares method of curve fitting.

Real data however, are a bit more complicated. In real data, each measurement is subject to random measuring errors, so a theoretically ideal model may never precisely describe the measurements. Fortunately, there is a way around the problem by performing a weighted fit. In a weighted fit, greater significance is given to the more accurately determined data points. All that is needed is some measure of the "goodness" of the fit. In least-squares fitting the figure of merit and this "goodness" measurement are the same quantity, but this is not always so as we demonstrate later in this section. In least-squares fitting the quantity to be minimized is a function $S(c_n,x_i)$ with n=1,...,M the number of constants in the fit and i=1,...,N the number of data points, where;

$$S = \sum_{i=1}^{N} \omega_i (f_i - f_{th}(c_n, x_i))^2 \quad \text{.....(4.1)}$$

with

x_i	=	the i'th channel of the spectrum
c_n	=	the constants for the fit
f_i	=	actual data value at channel i
f_{th}	=	theoretical model of the data determined using the constants c_n
ω_i	=	the weight on the data point at channel i.

In a weighted fit, the weighting for each individual channel must be specified. ROBFIT contains the following options for choosing the weights.

a) Weight = $\frac{1}{\text{Data value}}$; this is usually the case with gamma-ray spectra. Here, the assumption is that the deviation in a particular channel follows Poisson statistics.

b) Estimated from the spread in the data. The channel-to-channel fluctuations are used to give an estimate of the variance and hence, the weight, from the expression weight = $\frac{1}{\text{Variance}^2}$.

c) User defined at each channel. The code has the capability to read a file which contains the weights for each data point.

Each of the above weighting schemes is discussed in greater detail in the following sections.

If the data deviations are distributed about the mean position in a Gaussian manner, then the quantity S is equivalent to the "goodness" of fit estimator. This means that minimizing S can also be used to give the best parameters for the fit. For a more detailed account of least-squares fitting and an explanation of the χ^2 function, we refer the reader to a book by Bevington[11]. We discuss the treatment of non-Gaussian statistics in Section 4.1.2 and in Chapter 8, but for now we simply indicate that equation 4.1 is the quantity that is minimized. The "goodness" of the fit is also measured using equation 4.1. For an ideal fit, the value of S should be equal to the number of data points (N) minus the number of constants (M). The quantity N-M is called the number of degrees of freedom of the data. In statistical terminology, one says that the chi-square per number of degrees of freedom $\frac{\chi^2}{(N-M)}$ equals unity

for a good fit. Values greater than unity indicate a bad fit while values less than unity indicate an over estimation of the errors. Once the minimizing function S has been correctly formulated, an optimization procedure can be applied to extract the "best fit" parameters. ROBFIT uses the non-linear Newton-Raphson technique described in Section 4.3.

The following sub-sections describe how to choose the correct weights under various operating conditions.

4.1.1 The standard weighting scheme

Frequently the channel weights can be calculated directly from the individual data points. For example, in gamma-ray spectra, a data value corresponds to the number of measurements of a particular gamma-ray energy in a certain time interval. These measuring statistics follow a Poisson distribution. Each data point is a single measurement of the mean number of counts, f_i, for that interval. The standard deviation on this value is given by $\sigma_i = \sqrt{f_i}$. These deviations can be used to calculate the weights on individual channels from $\omega_i = \frac{1}{\sigma_i^2}$.

Problems can arise when dealing with small data values, where small means less than five counts per channel. In this situation, the weights must be chosen with care.

4.1.2 Dealing with low numbers of counts

When data containing count rates of about five or less per channel are analyzed, fluctuations in the data values can lead to large variations in the weights. In these regions we can still express the weights as

$$\omega_i = \frac{1}{\sigma_i^2}$$

where σ_i is the standard deviation in the counts at each channel i as described in Section 4.1.1. For moderately good statistics we can get an estimate of the weights directly from the data by setting the variance (σ_i^2) as

$$\sigma_i^2 = f_i \qquad \text{giving} \qquad \omega_i = \frac{1}{f_i}$$

where f_i is the data value at channel i. As the f_i values become low, these two equations start to break down, showing that the single data point can no longer be used as a good estimator of the standard deviation[18]. This can be illustrated by considering the data values to be

$$f_i = \overline{f_i} + \Delta(f_i)$$

where $\overline{f_i}$ is the expectation value of f_i as calculated from a large ensemble of measurements of f_i, and $\Delta(f_i)$ is the random statistical error in f_i. The correct variance would be

$$\sigma_i^2 = \overline{\Delta(f_i)^2} \qquad \text{with} \quad \overline{\Delta(f_i)^2} = \overline{f_i}$$

for Poisson statistics. We do not have a good estimate of $\overline{f_i}$ with only a single measurement. Taking the data value f_i as the $\overline{f_i}$ introduces large fluctuations in the weights. If the weighting criteria are not changed and a peak fitted to this low data using the weighted least-squares method, the fit will systematically underestimate the area of the peak. This can be explained by considering the positive and negative fluctuations of $\Delta(f_i)$. For positive fluctuations f_i is larger than $\overline{f_i}$ creating a small weight that will underweight these positive going values. For negative fluctuations f_i is smaller than $\overline{f_i}$ making the weight too large, leading to overweighting of the negative going values. The size of the fitted function for small counts is seen to be systematically biased to lower peak heights. This bias in the weighting is also true for the larger data values, but there the $\frac{1}{f_i}$ weighting is not as sensitive to small changes.

Several changes to the method of least-squares have been tested and found to give unbiased weights in the low statistics region[19]. The most successful methods use a smoothing algorithm to replace the actual data value with its smoothed estimate. This helps reduce the data fluctuations, enabling them again to be used in the weighting determination. ROBFIT uses a similar

method since it replaces the data value by the fitted value, or one, whichever is the largest. The fitted value has been calculated using the entire spectrum and provides a better estimate of the underlying function. As the code adds peaks and changes the background, the weights are continually updated. At the outset of peak fitting there is a tendency to misrepresent the weights under a peak. However, iterating on peak and background fitting, and redetermining the weights, ensures that the correct weights are eventually found. This process is illustrated in Figure 4.1.

4.1.3 What to do when the weights are unknown

With gamma-ray spectra the individual channel weights are simple to calculate. For spectra, in which the statistical nature of the data fluctuations are unknown, the channel weights can be estimated from the channel-to-channel variations in the data. Consider the following equations in which the background is stored as a function $f_b(x_i)$, and the foreground, or peaks, as $p(x_i)$. The fitted function can then be represented as

$$f_{th}(x_i) = f_b(x_i) + p(x_i)$$

The data fluctuations around the fitted curve are given by

$$d_i = f_i - f_b(x_i) - p(x_i)$$

d_i being the difference between the data and the fit shown in Figure 4.2. The usual definition of a weight for the i^{th} channel is given by

$$\omega_i = \frac{1}{\sigma_i^2}$$

where σ_i^2 is an estimate of the variance at the i^{th} channel. Generally, the variance of a set of measurements (m_j) is given by

$$\sigma_i^2 = \langle (m_j - \mu)^2 \rangle$$

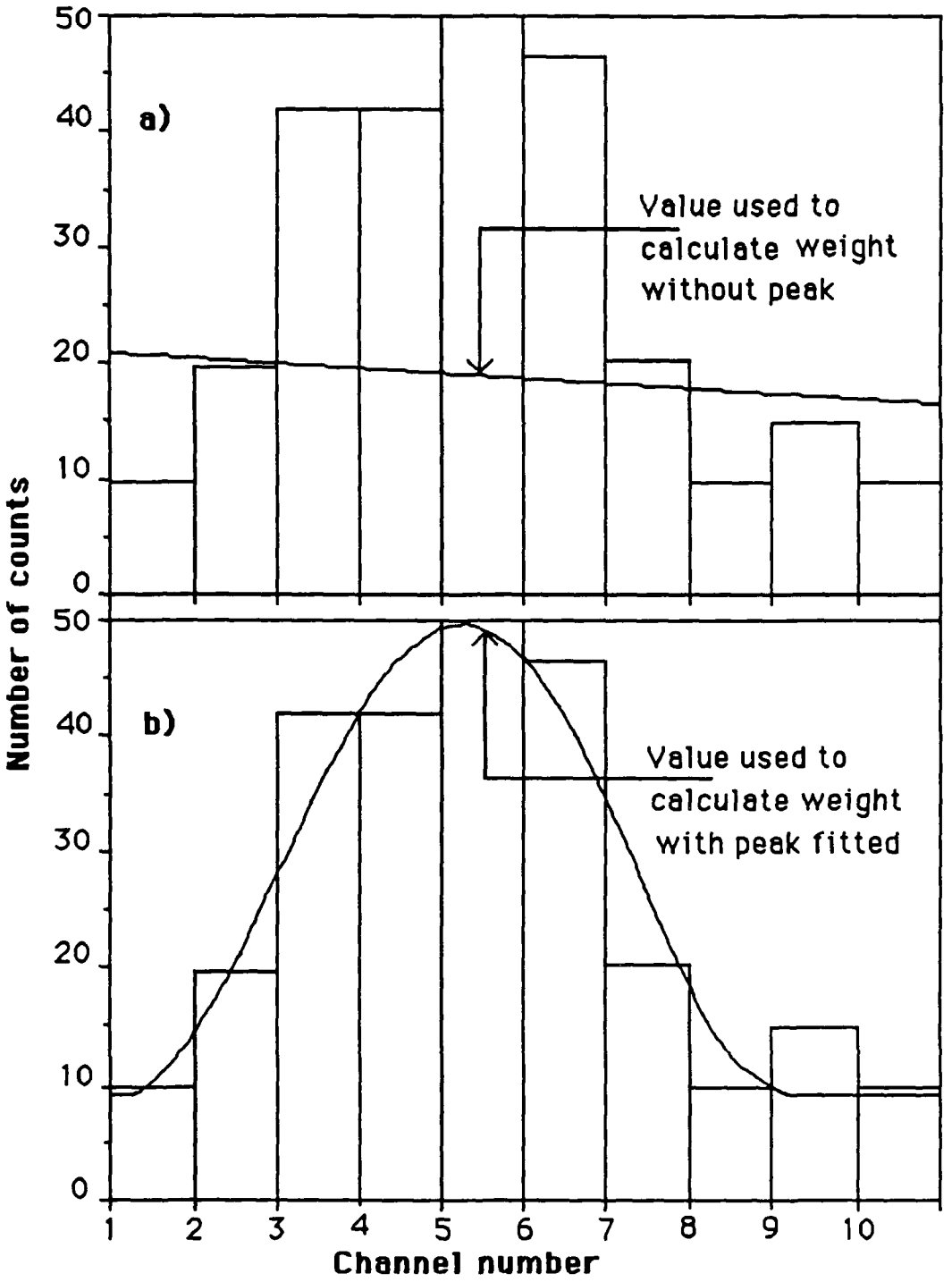


Figure 4.1 a) A fit consisting of a background function alone. The weight of channel 5 is calculated from the fitted value indicated. b) The effect of adding in the peak alters the fit so that now the weight is calculated correctly.

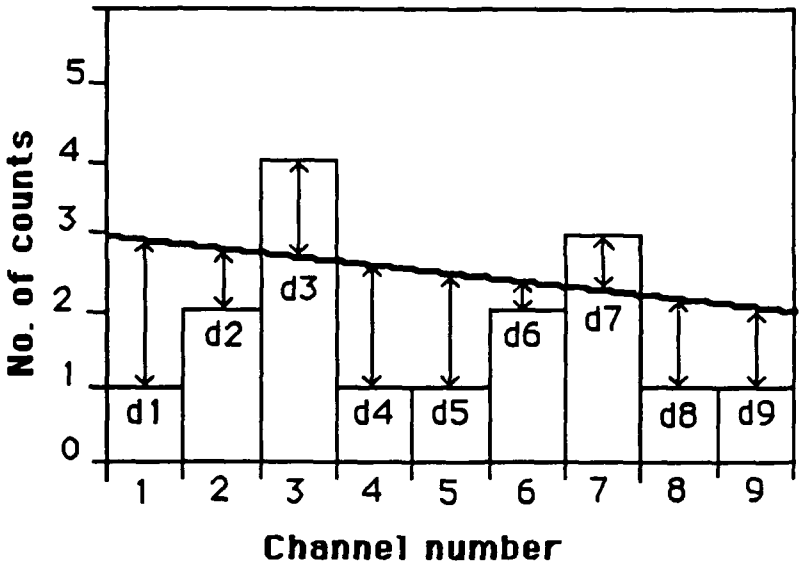


Figure 4.2 Data fluctuations around the fitted curve.

where μ is the mean of the parent distribution and $<>$ represents an expectation value. The expectation value of a function f(x) is defined as the weighted average value of the function averaged over all possible values of the variable x so that

$$<f(x)>=\lim_{N\to\infty}\left[\frac{1}{N}\sum_{i} f(x_i)\right]$$

Using the curve fit, at channel i, $f_b(x_i)+p(x_i)$ represents the mean, μ. The variance can, therefore, be expressed as

$$\sigma_i^2 = <d_i^2>$$

Once the fit has been established, the channel weights can be calculated using this equation. At the outset of fitting, however, the background and foreground are not well-defined and d_i must be calculated by an alternative method. An estimate of d_i, d_i', can be made by considering the differences on

either side of the i^{th} channel. Figure 4.3 shows that this estimate is given by

$$d_i' = 2f_i - f_{i+1} - f_{i-1}$$

and so

$$\langle d_i'^2 \rangle = \langle (2f_{th}+2d_i - (f_{th}+d_{i+1} + f_{th}+d_{i-1}))^2 \rangle$$

If the differences d_i, d_{i+1}, and d_{i-1} are randomly distributed about the mean, f_{th}, which is assumed constant across the three channels, then the expectation values for the cross products will average out to zero, leaving

$$\langle d_i'^2 \rangle = \langle 4d_i^2 + d_{i+1}^2 + d_{i-1}^2 \rangle$$

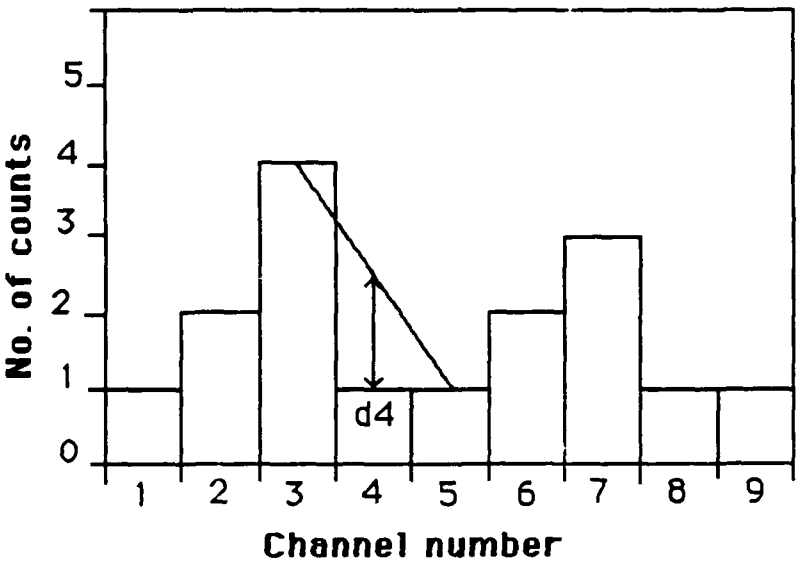


Figure 4.3 Estimating the difference at channel 4 at the outset of fitting.

From which we estimate

$$\langle d_i^2 \rangle = \frac{1}{6} \langle d_i'^2 \rangle \qquad \text{.....(4.2)}$$

A better estimate of the variance uses the fitted f_{th}, which is not quite constant across the three channels, to form

$$\sigma_i^2 = \frac{1}{6}<4(f_i-f_{th})^2 + (f_{i+1}-f_{th})^2 + (f_{i-1}-f_{th})^2> \quad .....(4.3)$$

and the weights can be estimated in the usual manner using

$$\omega_i = \frac{1}{\sigma_i^2}$$

At the outset of fitting, the weights are determined using equation 4.2. They are then continually updated during fitting using equation 4.3. This combination ensures the best possible weighting at each stage of fitting.

Before implementing this weighting scheme, a further correction is made to the weights. Within the range of a peak, where range is defined as twice the peak width (wp), there is a tendancy for the fitted function to follow the data. This is because the 2wp region is fitted with a three constant function (position, width, and strength of the standard peak), which results in over-fitting. This is more pronounced for small peak widths. To correct for this the weights are scaled by a factor

$$\frac{\text{No. of channels}}{\text{No. of channels - 3}}$$

where no. of channels = 2wp, giving the scaling factor

$$\frac{wp}{(wp-1.5)}$$

This correction compensates for the reduced degrees of freedom of these small regions.

4.1.4 Using the correct weights when subtracting spectra

ROBFIT gives the user the ability to subtract one spectrum from another. Subtraction is useful when comparing two spectra which differ by a small signal that is present in only one of the spectra. An example of the use of this technique is the analysis of the antarctic supernova data[1]. Here, two spectra

were effectively taken: the first, 'on' supernova, with the instrument pointing directly at the supernova and the second, 'off' supernova, with it pointing away. All non-supernova signals are then, ideally, contained within the 'off' spectrum while the tiny supernova signal is picked up in the 'on' spectrum. The 'off' spectrum contains the background to the supernova signal. This is not to be confused with the background continuum which resides under the peaks in both spectra. The procedure is to subtract the 'off' background from the 'on' spectrum, leaving the residual supernova signal.

Subtraction of spectra can be used to enhance a small signal as described above, but one must be cautious concerning its accuracy. After a subtraction, the random fluctuations from channel-to-channel in the residual spectrum will have increased. This can be shown by considering the subtraction of two numbers x_1, x_2 and their random errors $\Delta(x_1)$,$\Delta(x_2)$

$$(x_1 \pm \Delta x_1) - (x_2 \pm \Delta x_2) = (x_1 - x_2) \pm \sqrt{\Delta x_1^2 + \Delta x_2^2}$$

Here we see the error in the difference is larger than in the individual numbers. This is precisely what happens when two spectra are subtracted. This may mask the effects of a small peak. If the spectra contain only a simple underlying background continuum, generally, there is no gain from using the subtraction method. Small peaks can be identified by fitting the background continuum with few constants, and then removing its contribution from the spectrum. For a complex background, a subtraction helps only if the 'off' spectrum contains an adequate number of counts. When doing an experiment it is usual to spend most of the valuable observing time taking data for the 'on' spectrum. However, if it is the intention of the experiment that a subtraction is to be the way of analysis, then a reasonable amount of time must also be spent building up the 'off' spectrum. Otherwise, the poorly defined 'off' spectrum will dominate the errors of the subtraction. Therefore, there is a tradeoff between the number of constants to fit the background continuum and the number of counts recorded in the 'off' spectrum. Of course, this is dependent on the actual experiment being carried out.

ROBFIT can subtract spectra of differing normalizations. Two scaling factors are entered; one for each spectrum. The second spectrum is then re-

scaled by a normalization value equal to $\frac{\text{factor one}}{\text{factor two}}$. If these are unequal then an additional error proportional to $\left(\frac{\text{factor one}}{\text{factor two}}\right)^2$ is introduced. After subtracting the two spectra, the code stores the subtracted data and the sum of the spectra in two separate files. The summed data contains the individual channel weights for the subtracted data. These weights can be read by the main curve fitting code and applied to each channel when fitting the subtracted data.

Refer to Chapter 3 of Knoll[2] for a review of the statistical considerations involved in gamma-ray spectroscopy.

4.2 Overview of SMSQ the minimization algorithm

This section details the minimization algorithm, SMSQ, used to optimize the fit to the data. The object is to minimize χ^2, defined in Section 4.1, with respect to the peak and background constants. The required inputs into SMSQ are χ_0^2, the initial χ^2, $\frac{\partial\chi_0^2}{\partial c_l}$ and $\frac{\partial^2\chi_0^2}{\partial c_l \partial c_m}$, where the c's are the constants for the fit. Other inputs are Fr, the fractional reduction in χ^2 asked for in this minimization step, and $\mathbf{c_0}$, the starting values for the constants. In ROBFIT, the number of constants defining a fit can be anything from one to several hundred. To increase its fitting accuracy, knots are continually added to the background and new splines are added to represent peaks. Each background knot adds two constants (position and height) while each peak adds three (height, location and width). The time needed to fit the spectrum rises as the number of constants increases, so it is important to have a fast minimization scheme. ROBFIT uses a non-linear Newton-Raphson technique which is described as follows.

To recap, the quantity minimized is

$$\chi_0^2 = \sum_{i=1}^{N} \omega_i (f_i - f_{th}(\mathbf{c_0}, x_i))^2$$

where χ_0^2 is the initial value of χ calculated using $\mathbf{c_0}$. The procedure is to modify the constants $\mathbf{c}$ to reduce χ^2.

One minimization scheme would be to map out the χ^2 space for all possible values of the constants. For large numbers of constants, this method

becomes impractical. Consequently, ROBFIT uses an alternative method based on a non-linear minimization technique. The algorithms used are collectively called SMSQ. To find the optimum fit ROBFIT performs the following sequence of events.

a) At the start of minimization, SMSQ asks for a fractional reduction (Fr) in χ^2 equal to $Fr*\chi_0^2 (= \chi_p^2)$. To find this reduced value, SMSQ needs exact values of χ_0^2 and $\frac{\partial \chi_0^2}{\partial c_l}$ along with an approximate $\frac{\partial^2 \chi_0^2}{\partial c_l \partial c_m}$ for the M constants being fitted. These are used in the expansion

$$\chi_p^2 = \chi_0^2 + \sum_{l=1}^{M} \frac{\partial \chi_0^2}{\partial c_l}(c_{Frl} - c_{0l}) + \frac{1}{2}\sum_{\substack{l=1 \\ m=1}}^{M} \frac{\partial^2 \chi_0^2}{\partial c_l \partial c_m}(c_{Frl} - c_{0l})(c_{Frm} - c_{0m})$$

b) Using non-linear Newton-Raphson minimization and accelerated extrapolation techniques, SMSQ predicts the change of the constants, from c_0 to c_{Fr} needed to bring about the fractional reduction of step a). $\chi_p^2(c_{Fr})$ is the predicted value of χ^2 for these constants.

c) Using the predicted constants c_{Fr}, ROBFIT makes a new fit to the data and calculates $\chi_N^2(c_{Fr})$, the actual χ^2 for the c_{Fr} constants.

1. If $\chi_N - \chi_p <$ some tolerance level, the prediction is following the actual data and the fractional reduction can be made larger. Set $\chi_0 = \chi_N$, $c_0 = c_{Fr}$ and Fr=Fr*Fr.
2. If $\chi_0 < \chi_N$, then the step went too far. Leave χ_0 and c_0 unchanged, and set Fr=0.75+0.25*Fr.
3. If $\chi_p < \chi_N < \chi_0$, then the step is converging too slowly. Set $\chi_0 = \chi_N$, $c_0 = c_{Fr}$ and Fr=0.5+0.5*Fr. SMSQ repeats the above steps, from a), until either $\chi_p = \chi_0$, i.e.,

we can predict no lower value, or $\frac{(\chi_p - \chi_0)}{\chi_0} <$ some tolerance level. Additionally, a limit is placed on the total umber of loops the code can make in this minimization.

d) Once the minimum has been found, additional constants can be added to the fit and the whole process repeated from step a).

This sequence of events is shown in Figure 4.4.

The speed by which the code finds a minimum is directly related to the non-linear minimization step b), the step that calculates $\chi_p^2(c_{Fr})$. At the heart of the minimization scheme is a parameter, λ, somewhat akin to the Marquart parameter, that allows the predictor to rapidly converge on the $\chi_0^{2*}Fr$ reduction. A schematic of $\frac{\chi_p^2}{\chi_0^2}$ against $\log(\lambda)$, shown in Figure 4.5, illustrates how convergence is effected. For large λ, the change in the constants is small, and so χ_p approximately equals χ_0. For small λ, there can be large changes in the predicted χ. The object is to calculate the predicted χ_p at the Fr level. To home in on the turn in the curve, SMSQ decreases or increases λ until two points (A and B), which lie on either side of Fr are found. Accelerated extrapolation is then used to quickly calculate the $\chi_p(c_{Fr})$ value.

4.3 The χ^2 predictor

A brief outline of the SMSQ χ^2 predictor has been given in Section 4.2. Here we describe the mathematical basis of this non-linear Newton-Raphson minimization scheme.

There are two stages to finding $\chi_p^2(c_{Fr})$, the predicted-reduced χ_0^2. The first involves homing in on the $\chi_p^2(c_{Fr})$ region, which means finding values of χ_p^2 that lie above (χ_p^2A) and below (χ_p^2B) the $\chi_p^2(c_{Fr})$ value. The second stage involves extrapolation over the χ_p^2A to χ_p^2B range to find $\chi_p^2(c_{Fr})$. We will call stage one the non-linear Newton-Raphson minimization and stage two the accelerated extrapolation.

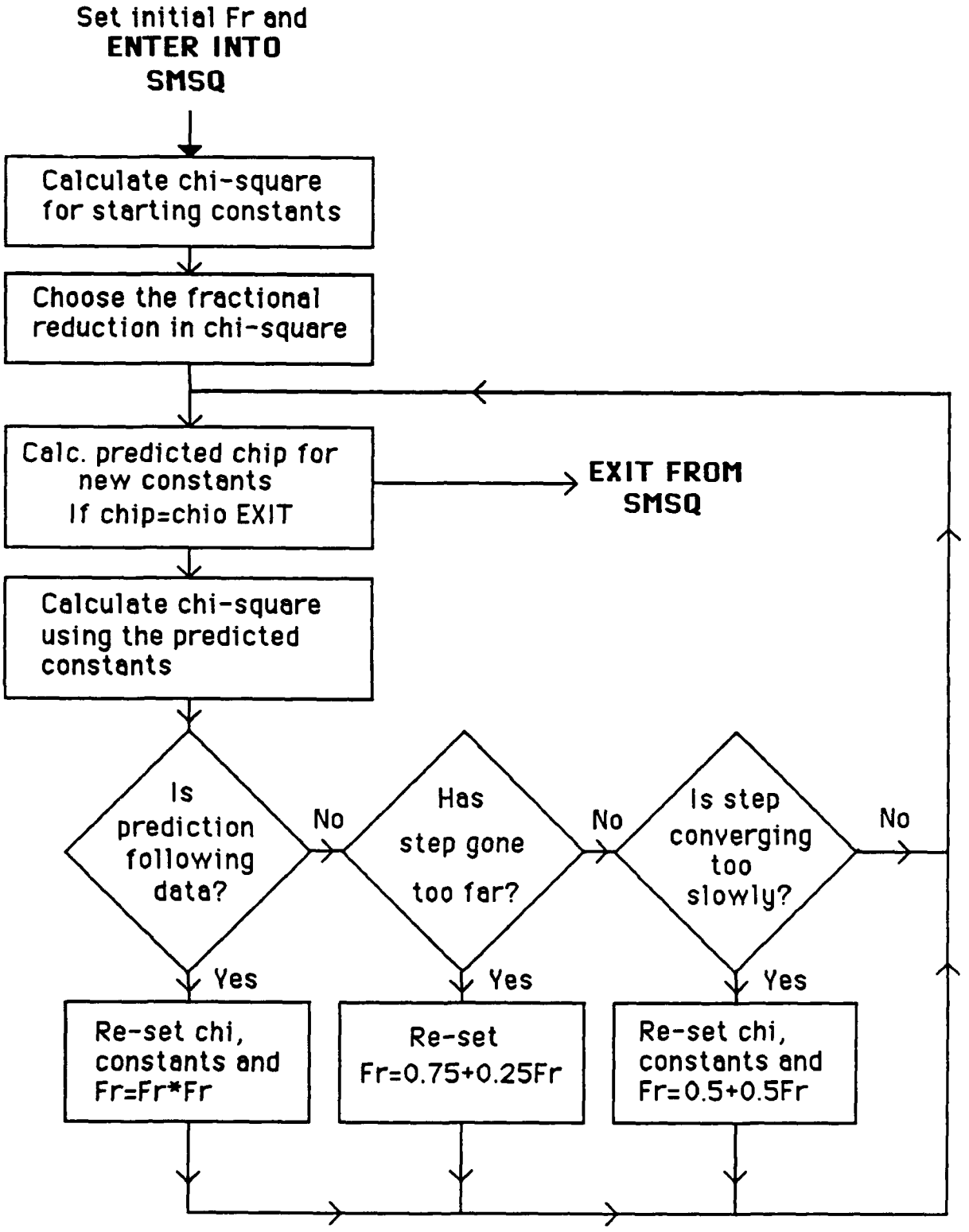


Figure 4.4 Flowchart of the minimization algorithm SMSQ.

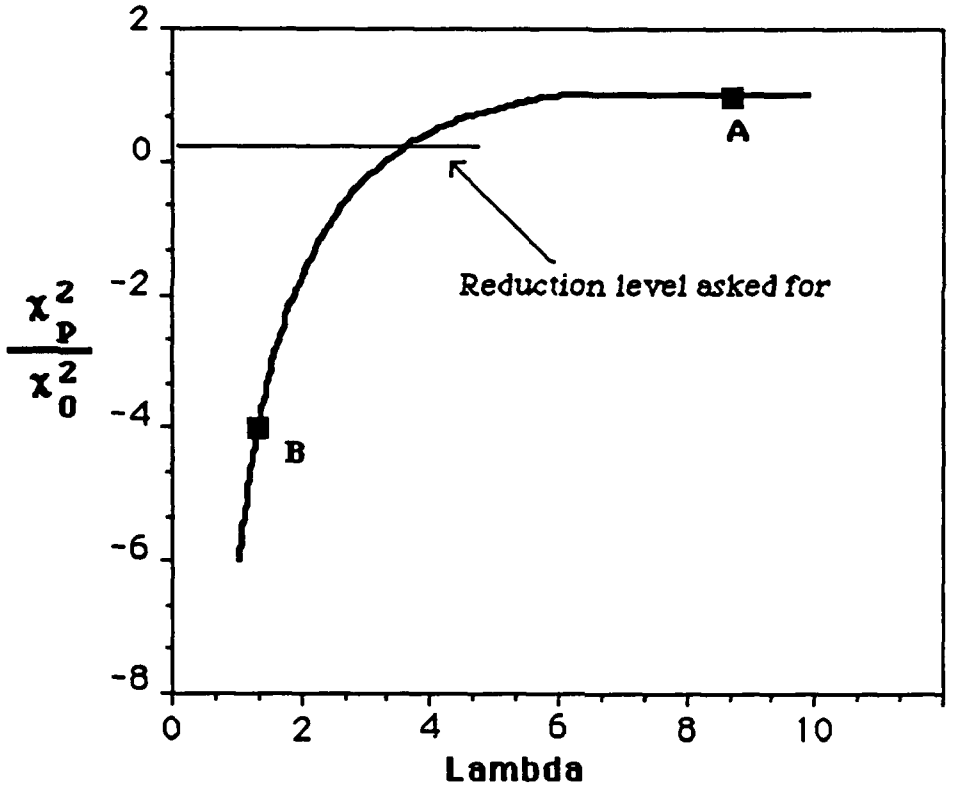


Figure 4.5 Plot of λ against the predicted χ^2.

4.3.1 Non-linear Newton-Raphson minimization

Before proceeding to discuss the SMSQ minimization routine, it is instructive to review the standard Newton-Raphson method. In Figure 4.6a the task is to find the point at which the function crosses the x-axis, which gives the root of the equation. At point x_1 the function value $f(x_1)$ and its derivative $f'(x_1)$ are used to predict the next point x_2, closer to the root of the function at x_r. At x_2, the function value $f(x_2)$ and its derivative $f'(x_2)$, are again used to predict a further point x_3, closer still to x_r. This procedure is repeated until the predicted point x_p is close to x_r. This method stems from the Taylor expansion of the function around a given point

$$f(x+\Delta) = f(x) + f'(x)\Delta + \frac{f''(x)\Delta^2}{2} + ...$$

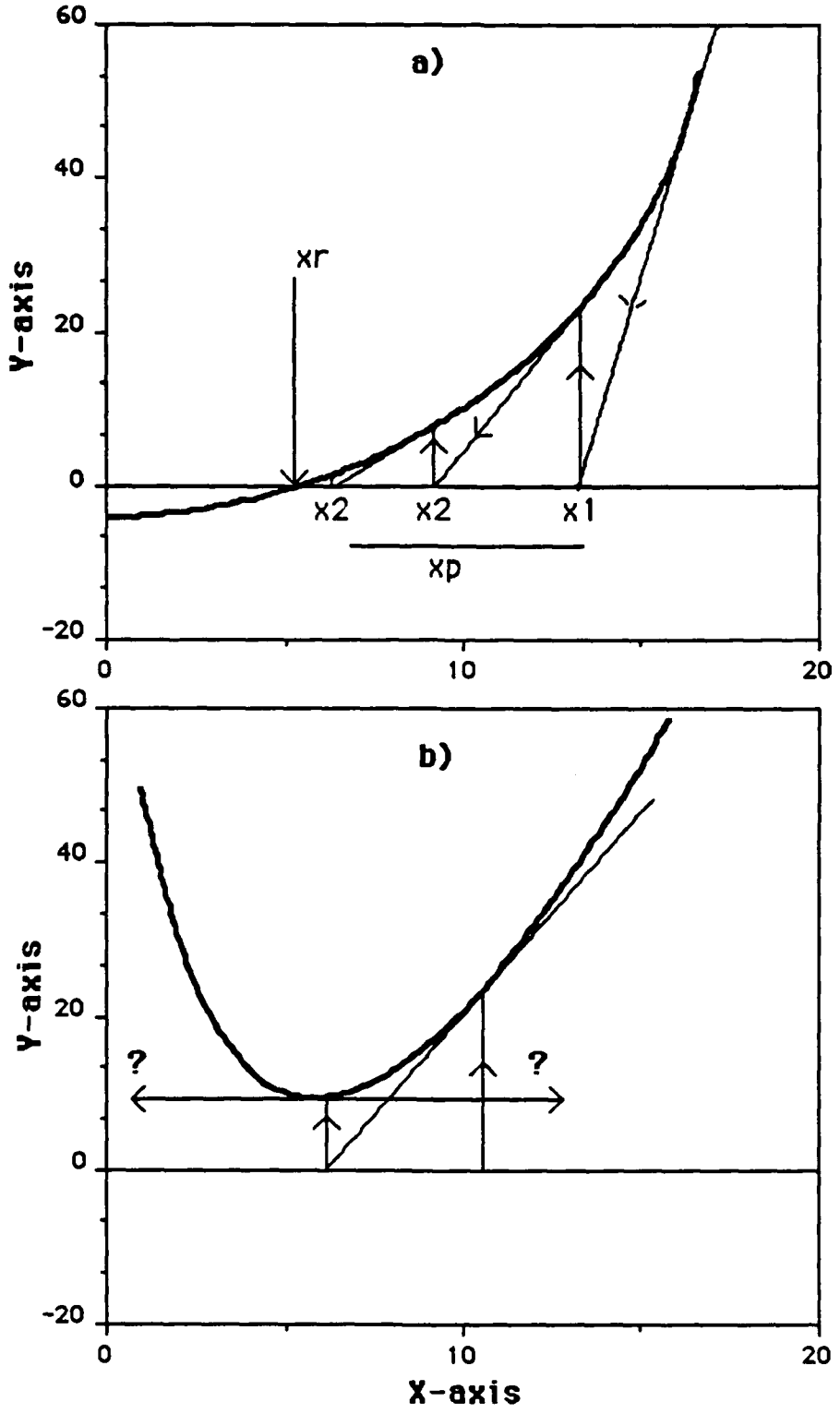


Figure 4.6 a) The standard Newton-Raphson technique, and b) problems that can occur if the curve contains a minimum.

For small values of Δ, terms of order Δ^2 or higher can be ignored. Around the x_r point, $f(x+\Delta)$ is approximately zero, giving

$$\Delta = -\frac{f(x)}{f'(x)}$$

the method then steps along the x-axis by the Δ amounts. This procedure, however, has problems when the function contains maxima and minima. Figure 4.6b illustrates this problem. If the predicted x_p happens to be close to either the maximum or minimum of the function, the next predicted step size can be extremely large. This makes the method unstable when dealing with complex functions containing many secondary maxima and minima.

To avoid this problem, ROBFIT uses a non-linear Newton-Raphson method. This method overcomes the large step size predicted in the above example by reducing the step size and trying again. Consider taking the function to be minimized to be the derivative of χ^2. Because we are looking for a minimum in the χ^2 function, if χ_p at the new point is predicted to be larger than the starting value of χ_0, then the step size is reduced. This re-evaluation of step size and calculation of χ_p continues until either χ_p becomes less than χ_0 or the step size is so small it becomes indistinguishable from the starting point. The step size is controlled by the parameter, λ, which can be increased or decreased in order to decrease or increase the step size. If χ_p is less than χ_0, then a larger step size can be used. The code continues to increase the step size to optimize the size of the final step. Varying the step size in this manner means that the procedure can rapidly move through flat regions, and accurately step through complex regions in the χ^2 parameter space. This sequence stops once the $\chi_p A$ and $\chi_p B$ of Figure 4.5 have been found. A flowchart of this sequence of events can be seen in Figure 4.7.

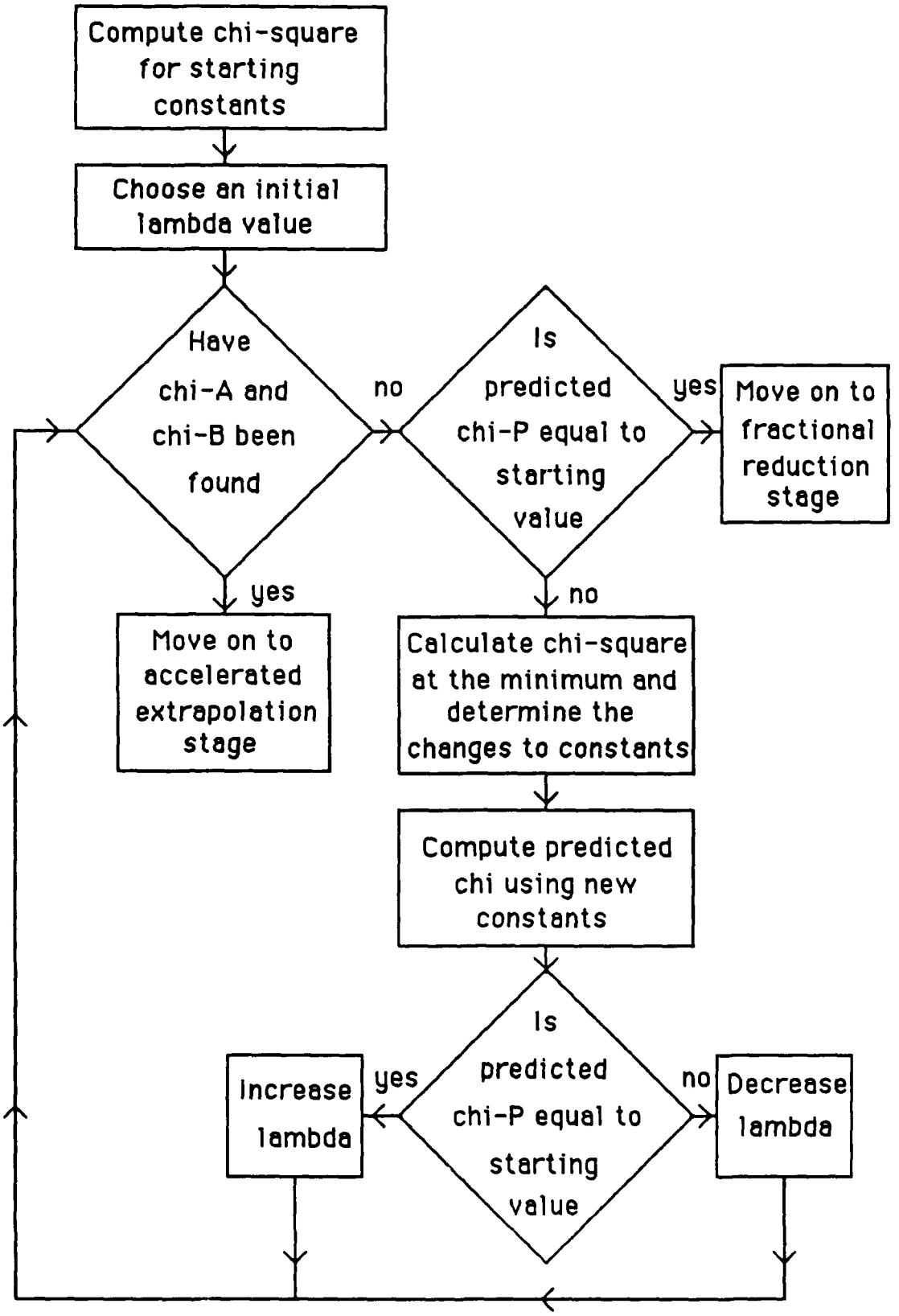


Figure 4.7 Flowchart of the procedures for calculating χ_A,χ_B

The following provides an introduction to the mathematics of the χ^2 iteration sequence. Consider the standard Newton-Raphson determination of χ_p

$$\chi_p^2 = \chi_0^2 + \sum_i \frac{\partial \chi_0^2}{\partial c_i}\Delta_i + \frac{1}{2}\sum_{ij} \frac{\partial^2 \chi_0^2}{\partial c_i \partial c_j}\Delta_i\Delta_j$$

where

$$\frac{\partial \chi_0^2}{\partial c_k} = -2\sum_i \frac{(f_i - f_{th}(\mathbf{c_0}, x_i))}{f_i} \frac{\partial f_{th}(\mathbf{c_0}, x_i)}{\partial c_k}$$

and

$$\frac{\partial^2 \chi_0^2}{\partial c_l \partial c_k} = 2\sum_i \frac{1}{f_i} \frac{\partial f_{th}(\mathbf{c_0}, x_i)}{\partial c_l} \frac{\partial f_{th}(\mathbf{c_0}, x_i)}{\partial c_k}$$

The parameter λ is then introduced so that the quantity to be minimized is

$$\chi_{Min}^2 = \chi_p^2 + \lambda \sum_i \Delta_i^2$$

The λ parameter corrects for the wild fluctuations that occur in the standard Newton-Raphson method. Differentiating we find

$$\frac{\partial \chi_{Min}^2}{\partial \Delta_l} = \frac{\partial \chi_0^2}{\partial c_l} + \sum_i \frac{\partial^2 \chi_0^2 \Delta_i}{\partial c_i \partial c_l} + 2\lambda\Delta_l$$

so that

$$\frac{\partial \chi_0^2}{\partial c_l} + \sum_i \left(\frac{\partial^2 \chi_0^2}{\partial c_i \partial c_l} + 2\lambda\delta_{il} \right)\Delta_i = \text{zero at the minimum}$$

where δ_{il} is the Kroenecker delta function.

From this equation Δ_i can be calculated using

$$\Delta_i = \frac{-\frac{\partial \chi_0^2}{\partial c_l}}{2\sum_i \left(\frac{\partial^2 \chi_0^2}{\partial c_i \partial c_l} + \lambda \delta_{il}\right)}$$

which are used to update the fitted constants,

$$c_0 = c_0 + \Delta_i$$

until the minimum has been found. The λ parameter can be adjusted, as previously described, to change the Δ_i values during fitting. Once $\chi_p A$ and $\chi_p B$ have been found, SMSQ moves to the accelerated extrapolation stage to find the value of χ_p at the χ_0^2*Fr level.

4.3.2 Accelerated extrapolation to find $\chi_p^2(c_{Fr})$

It is possible to continue the non-linear Newton-Raphson minimization, detailed in section 4.3.1, until the specified reduction in χ_0^2 has been achieved. Such a minimization scheme would be expensive in computer time. ROBFIT uses an accelerated extrapolation to speed up the calculation once $\chi_p A$ and $\chi_p B$ have been found.

The accelerated extrapolation method is called Aitken's delta-squared process after Aitken who first suggested it in 1926. The principle of the method is to solve an equation

$$f(x) = 0$$

by replacing it with an equivalent equation

$$x = g(x)$$

and computing a sequence using $x_{n+1} = g(x_n)$

$$x_1, x_2, x_3, \ldots x_n, \ldots . x_r$$

that converges on the root of f(x).

Suppose α is the root of f(x), then, provided $x_n \neq \alpha$

$$x_{n+1} - \alpha = g(x_n) - g(\alpha) = (x_n - \alpha)g'(\varepsilon_n)$$

with ε_n some point below x_n and α. Now

$$\lim_{n\to\infty} \frac{x_{n+1}-\alpha}{x_n-\alpha} = \frac{\partial g(x)}{\partial x}\Big)\alpha = g'(\alpha)$$

or equivalently

$$\frac{x_{n+1}-\alpha}{x_n-\alpha} = \frac{x_n-\alpha}{x_{n-1}-\alpha}$$

provided the value of g'(x) is approximately constant over the two steps. This gives a method for quickly calculating the root. For a function g which does not have a constant g', we may still write

$$\frac{x_{n+1}-x^p_{n+1}}{x_n-x^p_{n+1}} = \frac{x_n-x^p_{n+1}}{x_{n-1}-x^p_{n+1}}$$

and expect the value of x^p_{n+1} will be close to α. The number x^p_{n+1} is calculated from the expression

$$x^p_{n+1} = \frac{x_{n+1}x_{n-1}-x_n^2}{x_{n+1}-2x_n+x_{n-1}} \qquad \text{.....(4.4)}$$

Figure 4.8 illustrates how this accelerated extrapolation works in ROBFIT.

The procedure starts with the χ_pA and χ_pB points and uses the λ_A and λ_B as the x_n and x_{n+1} values. One further λ determination is required. The code keeps track of the χ_p calculated before χ_pA. For this purpose, the λ used to determine this χ is taken as the x_{n-1} value. Equation 4.4 is then used to make an estimate of the root x^p_{n+1}. Using this value of λ, a new χ_p is calculated. This is then taken as the x_{n+1} value and λ_A and λ_B are now used as the x_{n-1} and x_n points. This process is repeated until $\chi_p-\chi_0{}^2*Fr$ is within a predetermined limit, signifying that the predicted χ has been found.

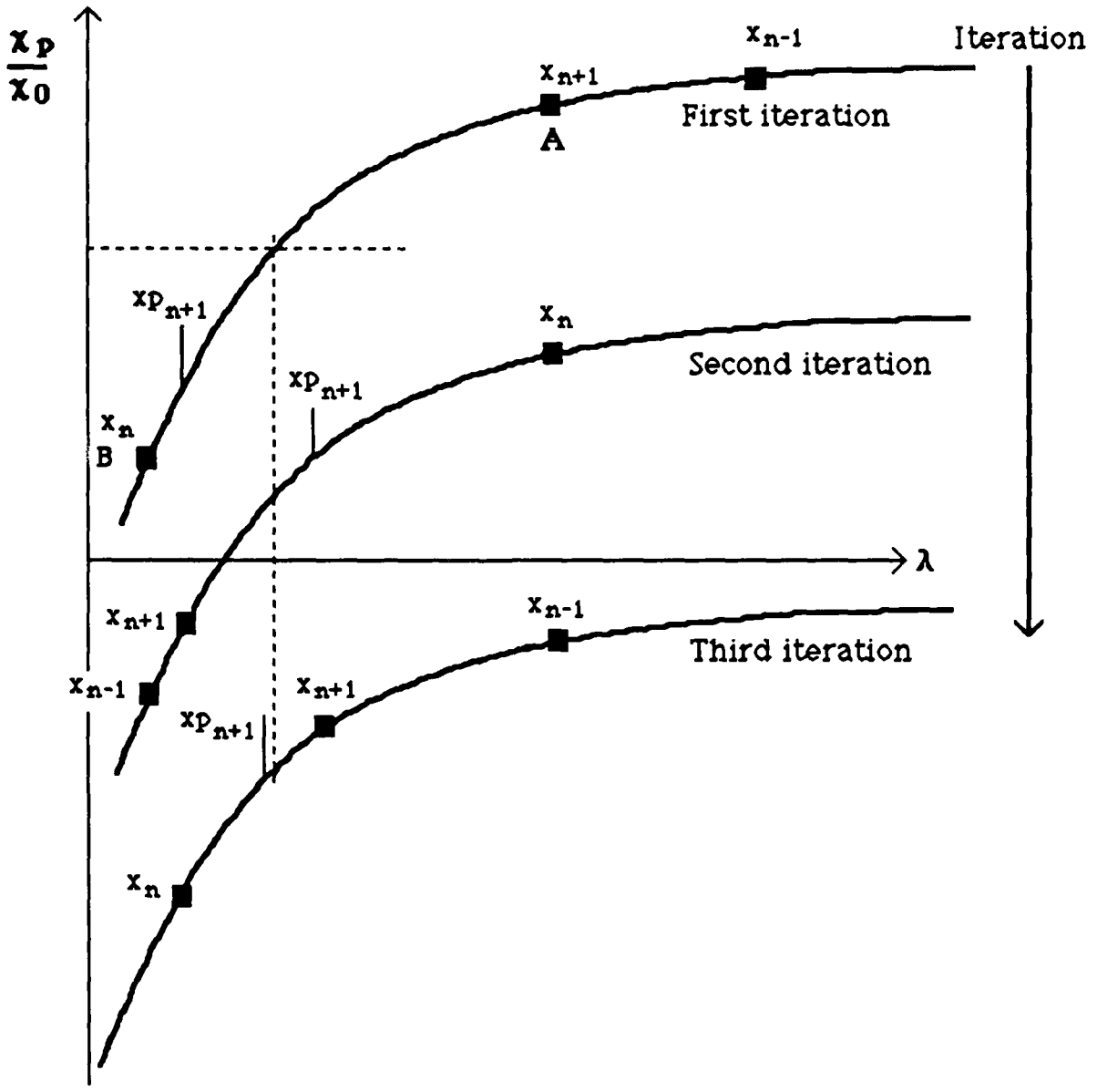


Figure 4.8 Iteration sequence of the accelerated extrapolation.

4.4 Using SMSQ outside ROBFIT

The SMSQ routines can be used as a general minimization package provided the operating parameters described in the previous sections are defined. Here we try to pull together the various aspects of the minimization procedure by showing how SMSQ can be used in a least-squares fit to artificially generated data samples. The following also introduces some of the more important points that need to be addressed when fitting histogram spectra.

4.4.1 Description of a non-linear least-squares fit routine that uses SMSQ

As usual, the quantity to be minimized is

$$\chi^2(c) = \sum_i \left(\frac{f_i - f_{th}(c,x_i)}{\varepsilon_i} \right)^2 \qquad \text{.....(4.5)}$$

where f_{th} is the function to be fitted to the f_i data points normally distributed about $f_{th}(c,x_i)$ with standard deviations ε_i. A routine called NLFIT is now used to perform the non-linear minimization. A complete listing of this routine is given in Appendix C. It assumes the data are in a file with one data point per line and written in free format with the form

$$\begin{array}{ccccc} x_{11} & x_{21} & x_{31} & f_1 & \varepsilon_1 \\ x_{12} & x_{22} & x_{32} & f_2 & \varepsilon_2 \\ .. & .. & .. & .. & .. \\ x_{1N} & x_{2N} & x_{3N} & f_N & \varepsilon_N \end{array}$$

where the dimension of the data, here three, the number of constants in the fit, and initial guesses at the constants; along with their relative nonlinearities, are read from another file which has the form

```
ADAT.DAT,       NAME OF INPUT DATA FILE
1,          NUMBER OF DIMENSIONS TO DATA X,F(X) IS ONE; X,Y,Z,F(X,YZ) IS 3
16,          NUMBER OF CONSTANTS BEING FITTED
nlfit.dir
   8776.88227475       1.00000000000      +- 395.
   1704.60713974       1.00000000000      +- 597.
  -906.526057399       1.00000000000      +- 318.
   219.659192191       1.00000000000      +- 84.0
  -30.0692438396       1.00000000000      +- 12.7
   2.52538578582       1.00000000000      +- 1.18
  -.135230639569       1.00000000000      +- 0.696E-01
  0.4619511977757E-02  1.00000000000      +- 0.260E-02
 -0.955385800160E-04   1.00000000000      +- 0.584E-04
  0.984502023374E-06   1.00000000000      +- 0.645E-06
 -0.100122518353E-09   1.000000000       +- 0.103E-14
 -0.762275403168E-10   1.0000000         +- 0.573E-10
  0.519858199369E-14   1.00000          +- 0.103E-14
  0.554323697394E-14   100.0000         +- 0.463E-14
  0,10000
  0,1000000
```

We call this file the direction file. The first derivatives of $\chi^2(c)$ are given by

$$\frac{\partial\chi^2}{\partial c_l} = -2\sum_i \left(\frac{f_i - f_{th}(c,x_i)}{\varepsilon_i}\right)\frac{1}{\varepsilon_i}\frac{\partial f_{th}(c,x_i)}{\partial c_l} \qquad \text{.....(4.6)}$$

while the second derivatives are adequately approximated by

$$\frac{\partial^2\chi^2}{\partial c_l \partial c_m} = 2\sum_i \frac{1}{\varepsilon_i}\frac{\partial f_{th}(c,x_i)}{\partial c_l}\frac{\partial f_{th}(c,x_i)}{\partial c_m} \qquad \text{.....(4.7)}$$

The term dropped in the second derivative is

$$-2\sum_i \left(\frac{f_i - f_{th}(c,x_i)}{\varepsilon_i}\right)\frac{1}{\varepsilon_i}\frac{\partial^2 f_{th}(c,x_i)}{\partial c_l \partial c_m}$$

The second derivative of f_{th} is hard to calculate accurately and is multiplied by a term which should average to zero. As long as this second term is small compared to equation 4.7, ignoring it will do no more than slightly slow the rate of convergence.

The exact form of f_{th} enters only in equations 4.5, 4.6 and 4.7. These are input in the form of

```
SUBROUTINE POLY(X,P,NV,ND,CONS,FA)
```

in which X(ND) is the array or x_i's, CONS(NV) are the current set of constants, FA is $f_{th}(c,x_i)$ and P(NV) are $\frac{\partial f_{th}(c,x_i)}{\partial c_l}$. In the event that the x_i is one dimensional and $f_{th}(c,x_i) = \sum_l c_l x^{l-1}$, this routine consists of

```
      P(1) = 1
      FA = CONS(1)
      DO 10 I = 2,NV
      P(I) = X(1)*P(I-1)
10    FA = FA + CONS(I)*P(I)
```

With a one dimensional x_i and f_{th} a spline of the form

$$f_{th}(c,x_i) = \sum_{L=1}^{4} c_L x^{L-1} + \sum_{L=5;7;...}^{NV} c_L (c_{L+1} - x_i)_+^3$$

where $(x)_+ = x$ for $x>0$

$= 0$ for $x<0$

the routine POLY consists of the above, together with

```
      DO 20 L = 5,NV,2
      P(L) = 0
      P(L+1) = 0
      XM = C(L+1) - X(1)
      IF(XM.GT.0) THEN
            P(L) = XM**3
            P(L+1) = -3*CONS(L)*XM**2
            FA = FA + CONS(L)*P(L)
      ENDIF
20    CONTINUE
```

The spline knots, C(L+1) for L>5, are non-linear. In the direction file they need to be made approximately 10^6 compared to the one approximate for the other constants. They also need to be specified reasonably close to their final values, else the routine will find an undesired local minimum rather than the desired minimum in χ^2.

The rest of NLFIT consists of rewinding the data file, calculating χ_0^2 and its partials from the values given by POLY, and calling SMSQ to find the new constants until the equivalence of χ_0^2 and χ_p^2 indicates a local minimum. The error treatment described in Chapter 7 is then used to convert a directly calculated inverse matrix into standard deviation estimates for the constants

and the values of $f_{th}(c,x_i)$. The routine writes a file containing

$$\begin{array}{cccccc} x_{11} & x_{21} & x_{31} & f_1 & f_{th}(c,x_1) & \sigma_1 \\ x_{12} & x_{22} & x_{32} & f_2 & f_{th}(c,x_2) & \sigma_2 \\ .. & .. & .. & .. & .. & .. \\ x_{1N} & x_{2N} & x_{3N} & f_N & f_{th}(c,x_N) & \sigma_N \end{array}$$

where σ_i is the standard deviation in the estimated value of the function.

4.4.2 Sample fits using NLFIT

Two sets of data were generated by randomly selecting points normally distributed about the straight lines extending from (0,10000) to (20,8000) to (40,8000) as shown in the following figures. The standard deviations of the random points from the lines were made equal to the square root of the value at the line, as would be expected in typical spectra. The straight lines in this case are the actual function. The simulation depicted is similar to the build up towards the Compton edge on the low energy side of a large gamma-ray peak. The object is to be able to resolve small peaks which include approximately five data points in a row, the sum of which is two to three standard deviations above the background shown, without introducing an excessive number of spurious peaks.

The area of a small peak at channel N is given by

$$A = \sum_{i=N-2}^{N+2} f_i - \sum_{i=N-2}^{N+2} f_B(x_i)$$

The standard deviation of the first sum is given by

$$\sigma_s^2 = \sum_{i=N-2}^{N+2} \varepsilon_i^2 = \sum_{i=N-2}^{N+2} f_i = S$$

while that of the second is approximately given by ε_i^2 at each point times the maximum of $\frac{\chi^2}{(N-M)}$ and 1 divided by $\left(\frac{N}{M}\right)$, the number of constants per region.

$$\sigma_B^2 = \sum_{i=N-2}^{N+2} \varepsilon_i^2 * \frac{M}{N} * \max\left(1, \frac{\chi^2}{N-M}\right)$$

Thus, the total error in A is given by

$$\sigma_A^2 = S\left(1 + \frac{M}{N} \max\left(1, \frac{\chi^2}{N-M}\right)\right)$$

where χ^2 is given by

$$\chi^2 = \sum_{i=1}^{N} \left(\frac{f_i - f_{th}(x_i)}{\varepsilon_i}\right)^2$$

and is a measure of the quality of our fit. Note that because of the maximum function, there is no reason for finding $\chi^2 < N-M$.

The sensitivity to small peaks is given by the ratio

$$R = \frac{A}{\sqrt{S}\sqrt{\left(1 + \frac{M}{N} \max\left(1, \frac{\chi^2}{N-M}\right)\right)}} \qquad \text{.....(4.8)}$$

When R for the number of channels, equal to the full width at half maximum for a given peak type, becomes less than the user defined CUTOFF, ROBFIT assumes that all detectable peaks have been found. Note that for M<<N, which is usually the case, the second square root can be set to one.

In the above, it has been assumed that $f_{th}(x)$ is randomly displaced from the true function. Fits to our artificial sets of data using cubic polynomials with four adjustable constants are shown in Figure 4.9. Both of these fits have a χ^2 of approximately 65 for 40 data points which makes the second square root equal to

$$\sqrt{1 + \frac{4}{40}\frac{65}{36}} = 1.09$$

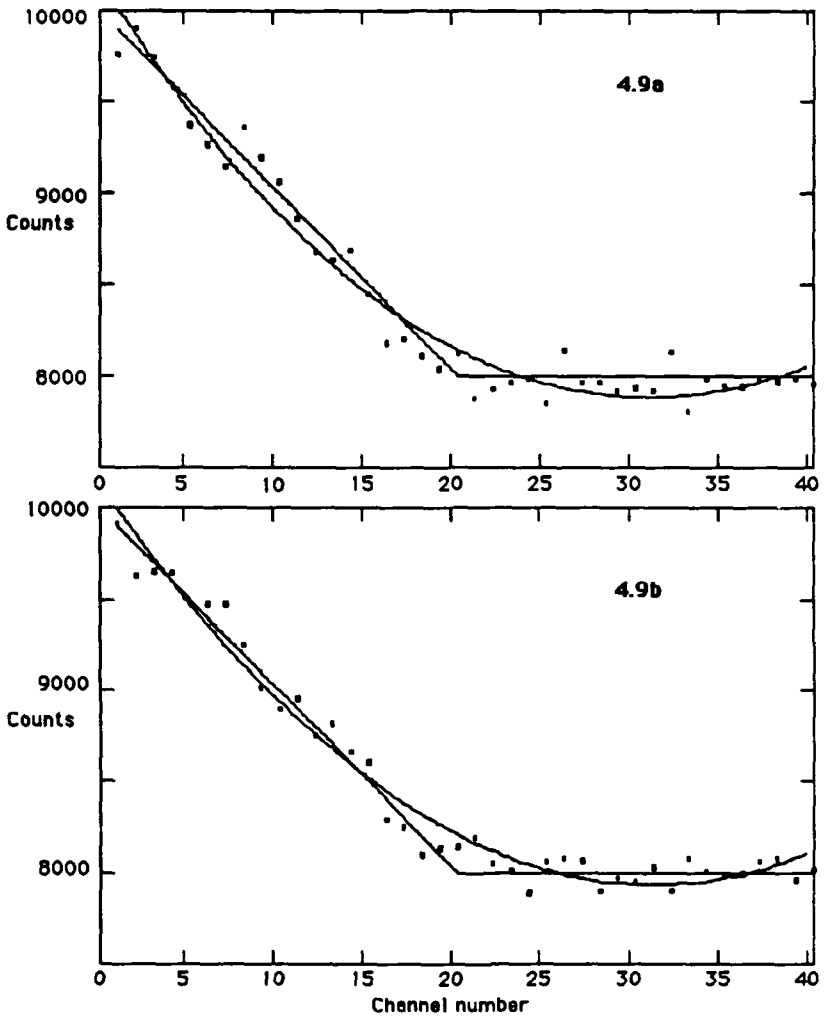


Figure 4.9 a) and b) Two, 4 constants polynomial fits. The straight lines are the original functions. The points are the randomized channel counts and the curved line is the fitted function.

which barely changes R. Both fits are too high by approximately 2σ in the region near channel 20, and too low by about 1σ in the regions near channels 10 and 30. This is despite the fact that we would expect the error in the fit to be

$$\sigma_B = \sigma \sqrt{\frac{4}{40}} * \sqrt{\frac{65}{36}} = 0.426$$

The extra error occurs because the polynomial is incapable of fitting the true function. The group of points near channel 10 in Figure 4.9a has an R given by

$$R = \frac{6\sigma}{\sqrt{5}\sigma} = 2.68$$

which means that it will be introduced as a spurious peak at above the 2σ confidence level. It will also reappear in the independent data of Figure 4.9b. In addition, real peaks will need to reach almost to the 5σ level before they will be detected near the data bend at channel 20. A better representation of the background is needed!

The most obvious solution is to simply add more constants, as shown in Figure 4.10 where the data have now been fitted to 5^{th} order polynomials with 6 constants. This gives a χ^2 of approximately 42. The spurious peak at channel 10 in Figure 4.9a moves to about channel 15 in Figure 4.10a with an R given by

$$R = \frac{4\sigma}{\sqrt{5}\sigma} = 1.79$$

which means that it will not normally be detected. The point at channel 26 in Figure 4.10b will also have about the same value for R and should not be detected. Both fits, however, round the corner at channel 20 and the χ^2 of 42 is definitely above the 34 possible. The question is, will more constants help? Fits to 9^{th} order polynomials using 10 constants are required to make $\chi^2 = 30$ (40 points - 10 constants), which is the theoretical limit. These are shown in Figure 4.11.

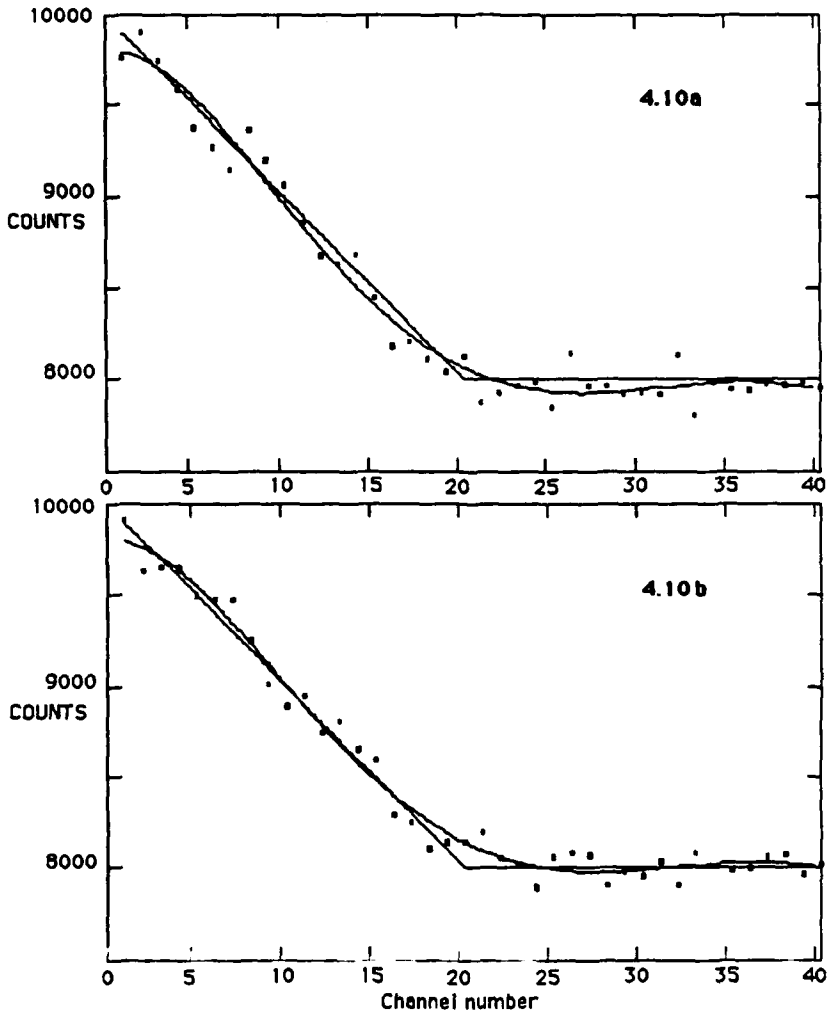


Figure 4.10 a) and b) Two, 6 constants polynomial fits.

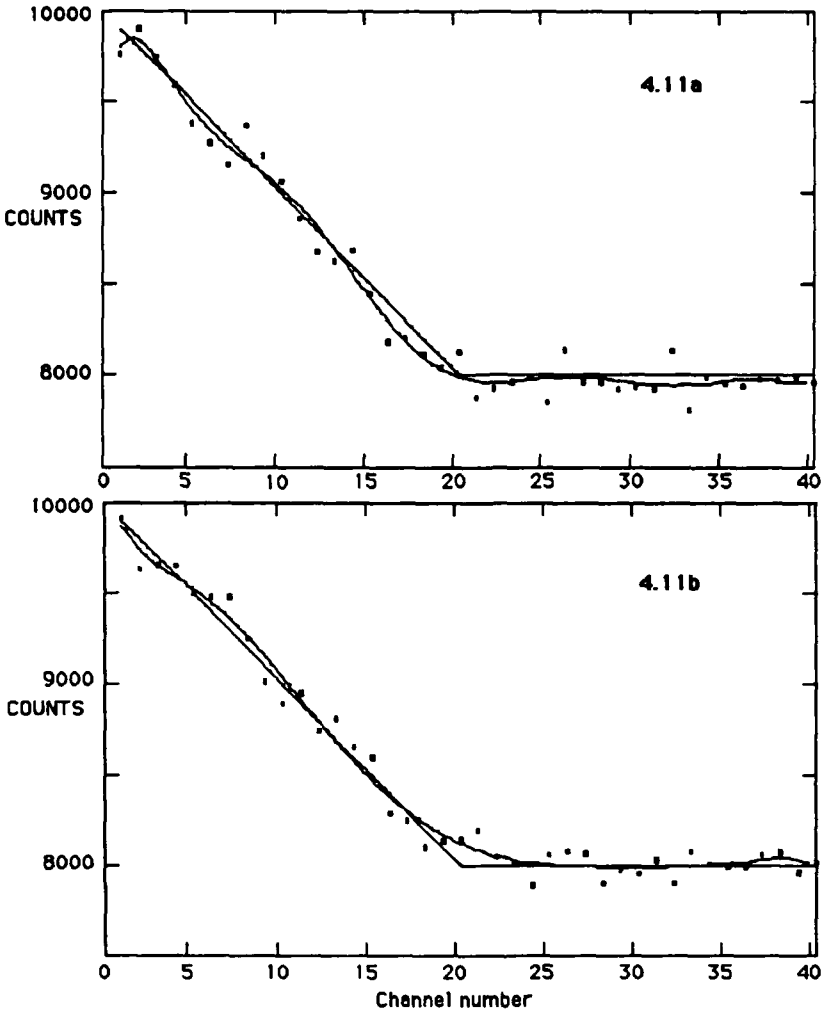


Figure 4.11 a) and b) 10 constant polynomial fits to the two data sets.

Additional constants do not introduce spurious peaks, but they do exhibit the well known phenomenon of looping through the data. This could hide peaks by reducing R in the peak neighborhood.

Adding constants to the fit can be carried to an extreme as shown in Figure 4.12 where 13^{th} order polynomials are fitted to the data. Each constant now tends to reduce χ^2 by 1, but not continuously. Note that in Figure 4.12b the fit now shows the possibility of a peak even at channel 22. The fit is over-following the data, but that is not all bad.

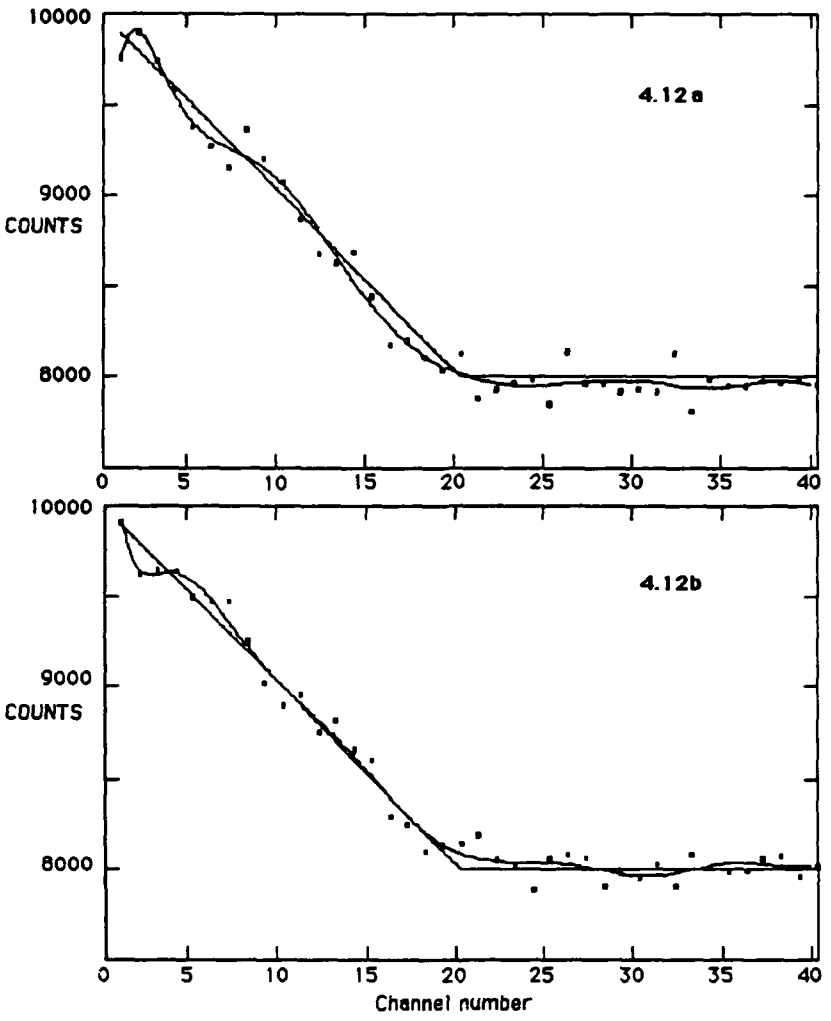


Figure 4.12 a) and b) The limit of polynomial fitting for this function. These fits have been performed with 14 constants.

The looping tendency is caused in part by trying to keep all derivatives continuous. The solution is to use splines in the fit. In Figure 4.13 the data has been fitted using fixed knot splines

$$f_{th}(x) = \sum_{i=0}^{3} c_i x^i + \sum_{i=1}^{6} d_i (\xi_i - x)_+^3$$

where

$$\xi_i = i * 6\frac{2}{3}$$

and the 10 c's and d's were determined by NLFIT.

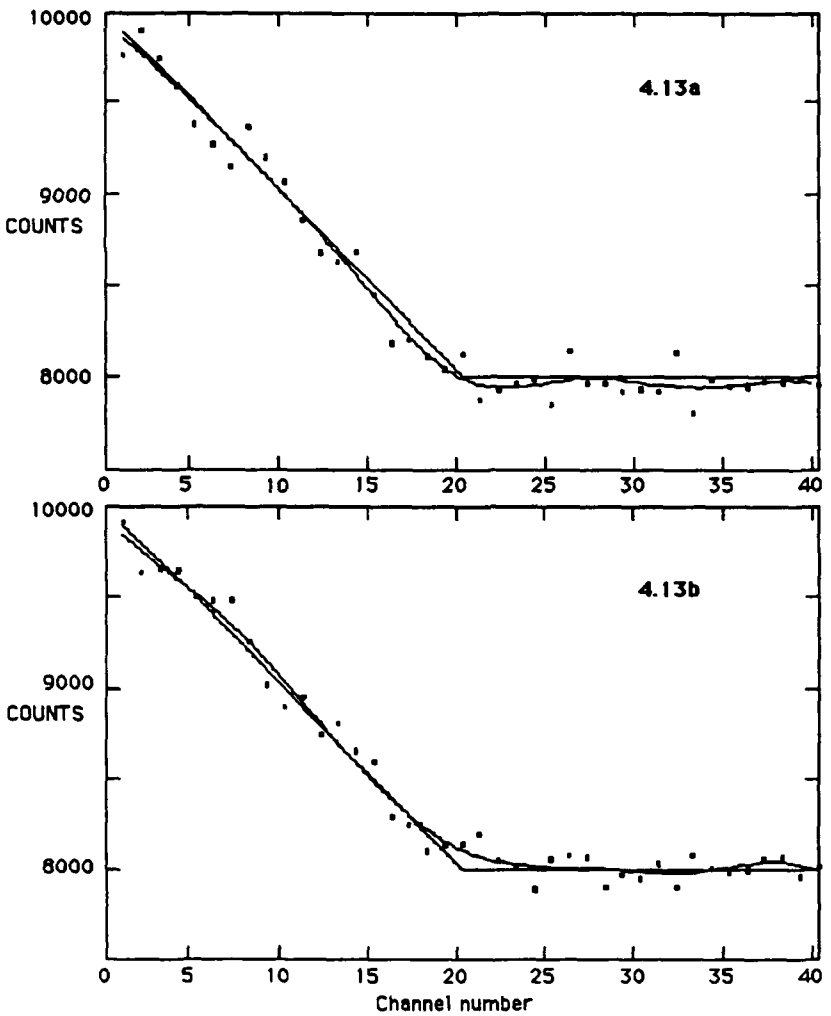


Figure 4.13 a) and b) 10 constant, fixed knot, spline fits to the two data sets.

In comparison with the 10 constant polynomial, the match to the data is better, but still not perfect. It is worth noting that NLFIT handles constants in this form much better than in the polynomial form. The fit with 10 knots (14 constants) is shown in Figure 4.14 where again the data is followed too closely. In Figure 4.14a χ^2 = 26, which is high for a fit containing 16 constants, while in Figure 4.14b χ^2 = 21, which is low. The expected standard deviation in χ^2 is $\sqrt{2N}$, or approximately 9 for 40 data points, indicating that these fits are consistent with one another. The fixed knot splines are a little easier to work with than the polynomials and allow us to use more constants to fit the data, but do not appear quantitatively better.

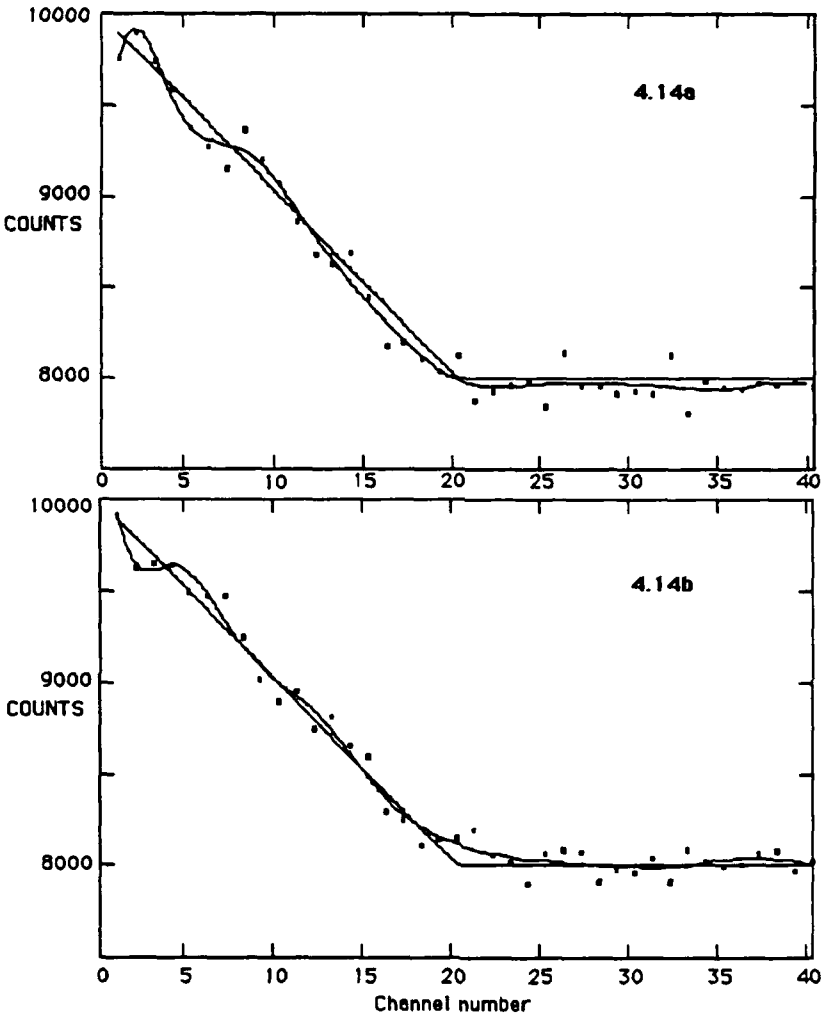


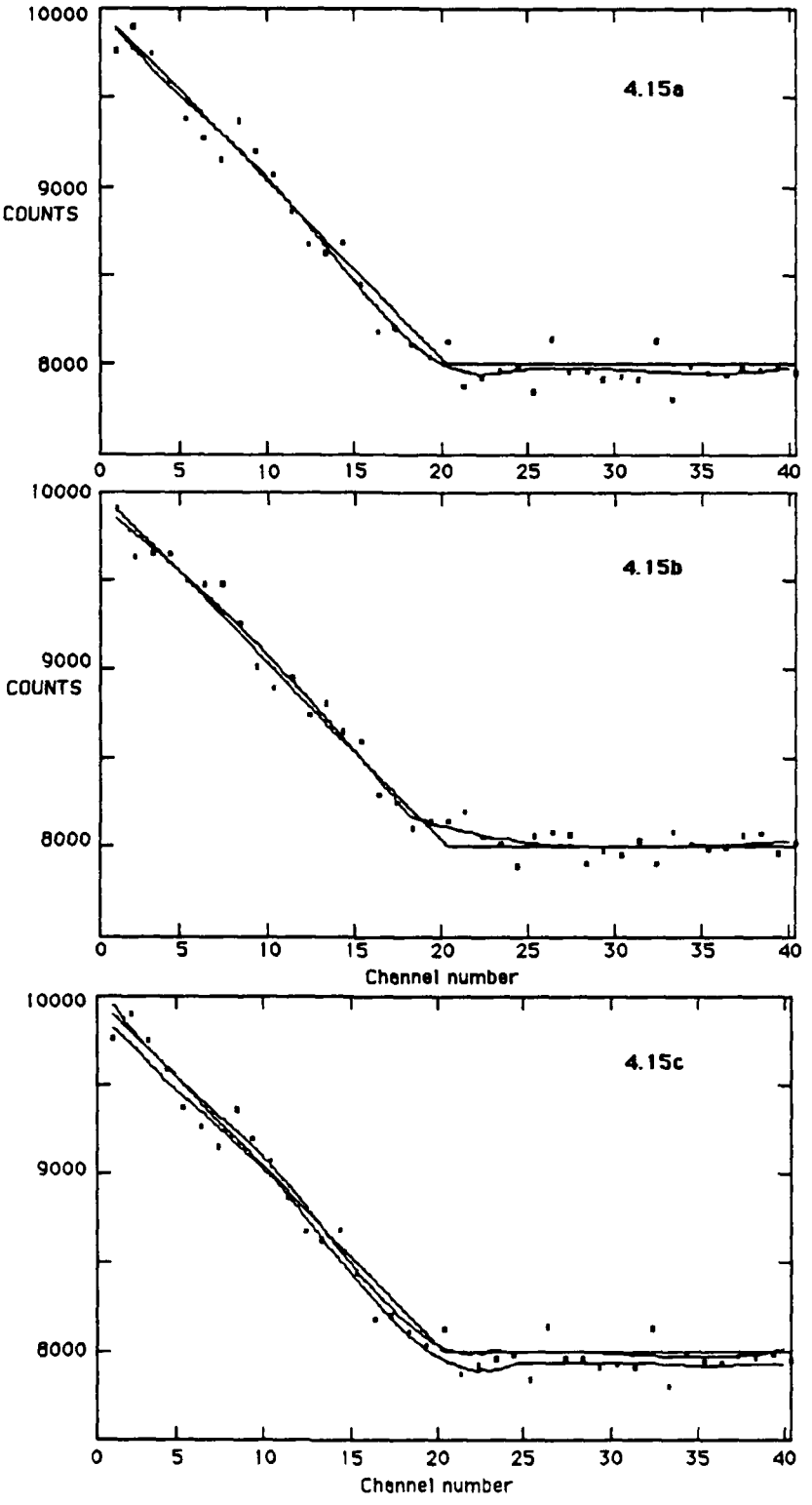
Figure 4.14 a) and b) 14 constant, fixed knot, spline fits to the two data sets.

The code NLFIT is capable of using splines with variable knots, which makes the fitting function non-linear

$$f_{th}(x) = \sum_{i=0}^{3} c_i x^i + \sum_{j\,by\,2's} c_j (c_{j+1} - x)_+^3$$

This dramatically slows NLFIT since it now takes 10's to 100's of steps to optimize with respect to the knot positions. In addition, there are now local minima so that one does not always find the lowest possible χ^2. When adding knots, the new knot is introduced close to, and on the higher channel number side of the largest residual. The knots were added one at a time to the best fit obtained without the knot. The result using 10 coefficients (three adjustable knots), is shown in Figures 4.15. Figure 4.15c illustrates the $+1\sigma$ and -1σ confidence limits for the fitted function of Figure 4.15a. These limits have been calculated using the error treatment discussed in Chapter 7. The knots are now involved in rounding the corner at channel 20, the spline with 14 parameters (five adjustable knots) is still reasonably smooth as shown in Figure 4.16, compared to Figure 4.12 for polynomials and Figure 4.14 for fixed knot splines. Too many constants always follow the data too closely as shown in the 16 parameter, six adjustable knot fits of Figure 4.17.

The conclusion is that too few constants used to fit the background can introduce regions where spurious peaks will be introduced. However, too many constants will cause the background to approach the peaks too closely, and thereby mask some of them. Polynomials are inherently limited to short ranges of data, and numerical precision stopped us from going beyond 13^{th} order in these tests. Fixed knot splines were fitted as quickly as polynomials, and in addition, worked for larger number of constants. Variable knot splines came closest to reproducing the underlying curve, as seen in Figures 4.15a through 4.15c, and also have the most tolerance for excess constants. The routine ROBFIT uses variable knot splines when *'CONT'* is specified in the startup menu and fixed knot splines when *'FIXK'* is specified. In each case the knots are added two at a time; one at the largest residual and the other, $\frac{N}{M}$, higher in channel number.



Figures 4.15 a) and b) 10 constant, variable knot, spline fits to the two data sets, c) The region of error on the 10 constant, variable knot, spline fit (a) above.

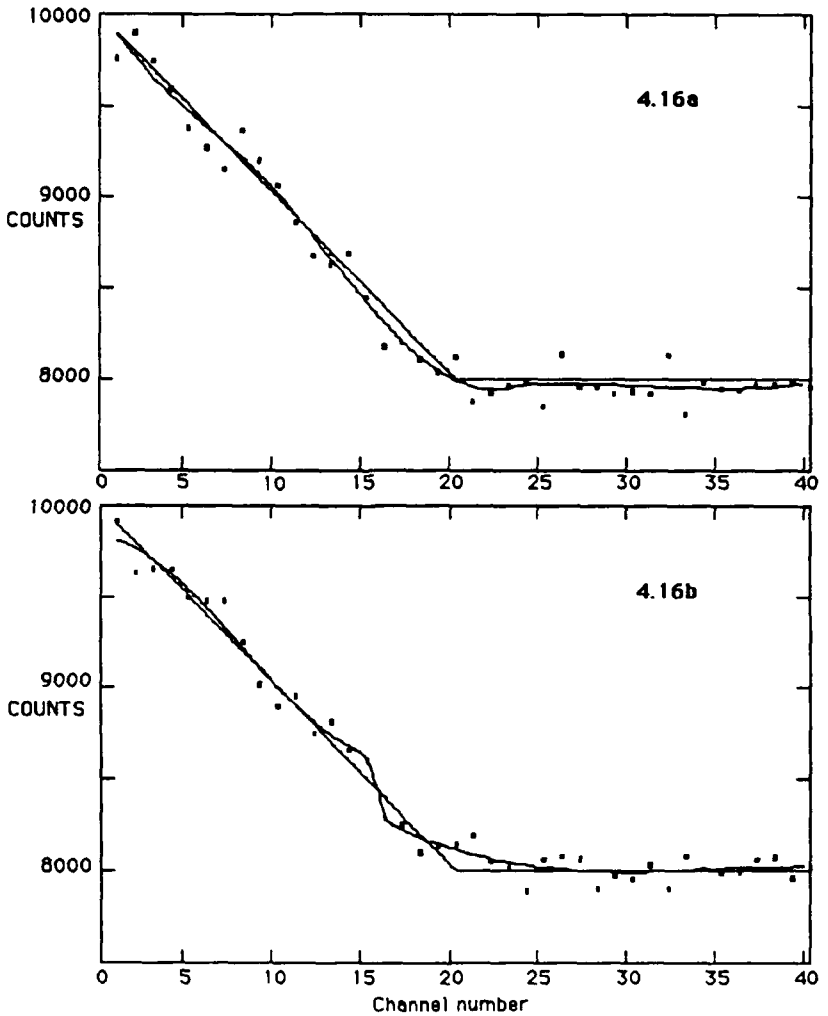


Figure 4.16 a) and b) 14 constant, variable knot, spline fits to the two data sets.

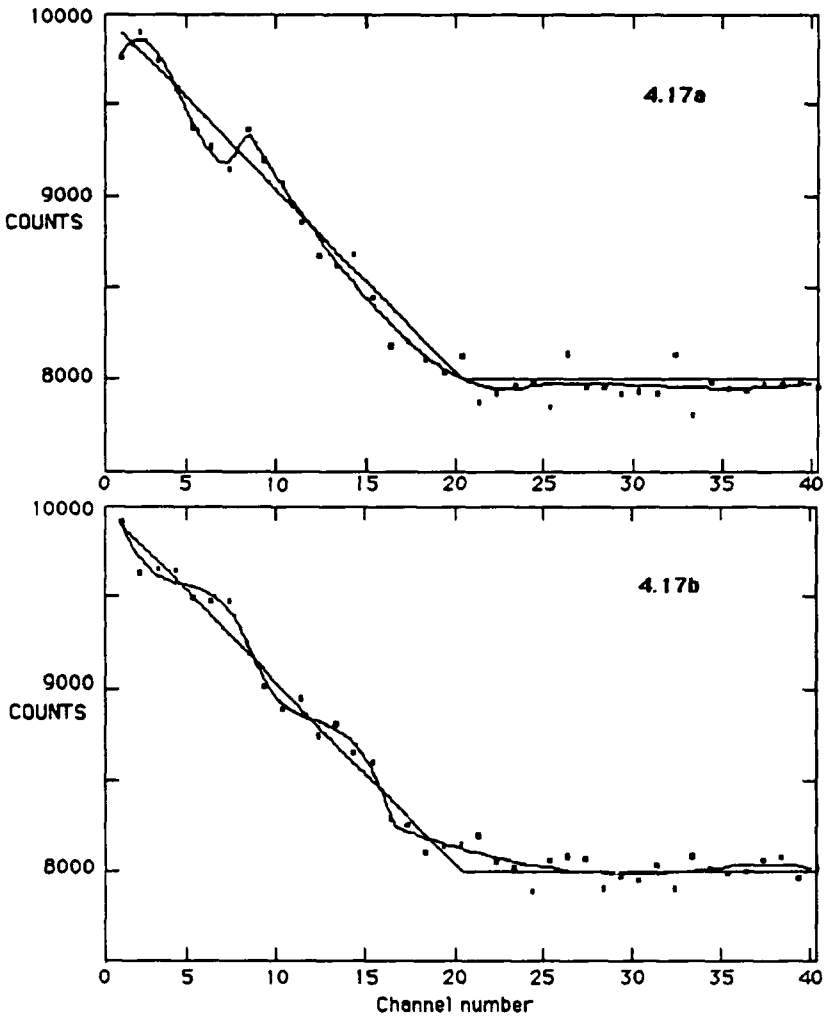


Figure 4.17 a) and b) The effect of over fitting the data with 16 constant, variable knot, spline fits.

5 Representation of the background

In this chapter we introduce the reader to the techniques used in fitting the background. The first section describes how robust estimation smooths the raw data and speeds up the background fitting process. In the second section we show how this smoothed background is fitted with splines. The final section contains the details of how the knots are moved when fitting these splines.

5.1 Robust fitting of the background

This section shows how ROBFIT implements robust fitting in the background determination. Various forms of robustness have been defined in relation to statistical estimation (see Press[20]). The general technique is to make a particular analysis insensitive to points that are not accurately defined. Consider Figure 5.1 for example, which shows how a linear least-squares fit to a straight line can be disrupted by spurious outlying points. Robust fitting gets around the problem by weighting down the points with large errors.

We use a similar weighting technique within ROBFIT to reduce the contribution from peaks during fitting of the background. To a certain level, until all peaks have been found, the background will contain some peak contamination. The object of the robust fitting is to enable a background fit to be performed even in the presence of this peak contamination. Within the fitting, we can bring this about by biasing against 'high lying' points while biasing for 'low lying' points. The code effectively seeks out the low regions within a spectrum, as illustrated in Figure 5.2.

Problems can occur if the data contains channels with zero or unusually low counts. Possible causes of these low values are data drop out or data corruption during the experiment. These low count regions are usually not related to the background function, and will pull the background fit down if they are not corrected. ROBFIT contains an option for recreation of data channels. The code linearly interpolates across the problem regions between two user chosen channels, and successively replaces the data values with interpolated values. All anomalous low count regions must be corrected before

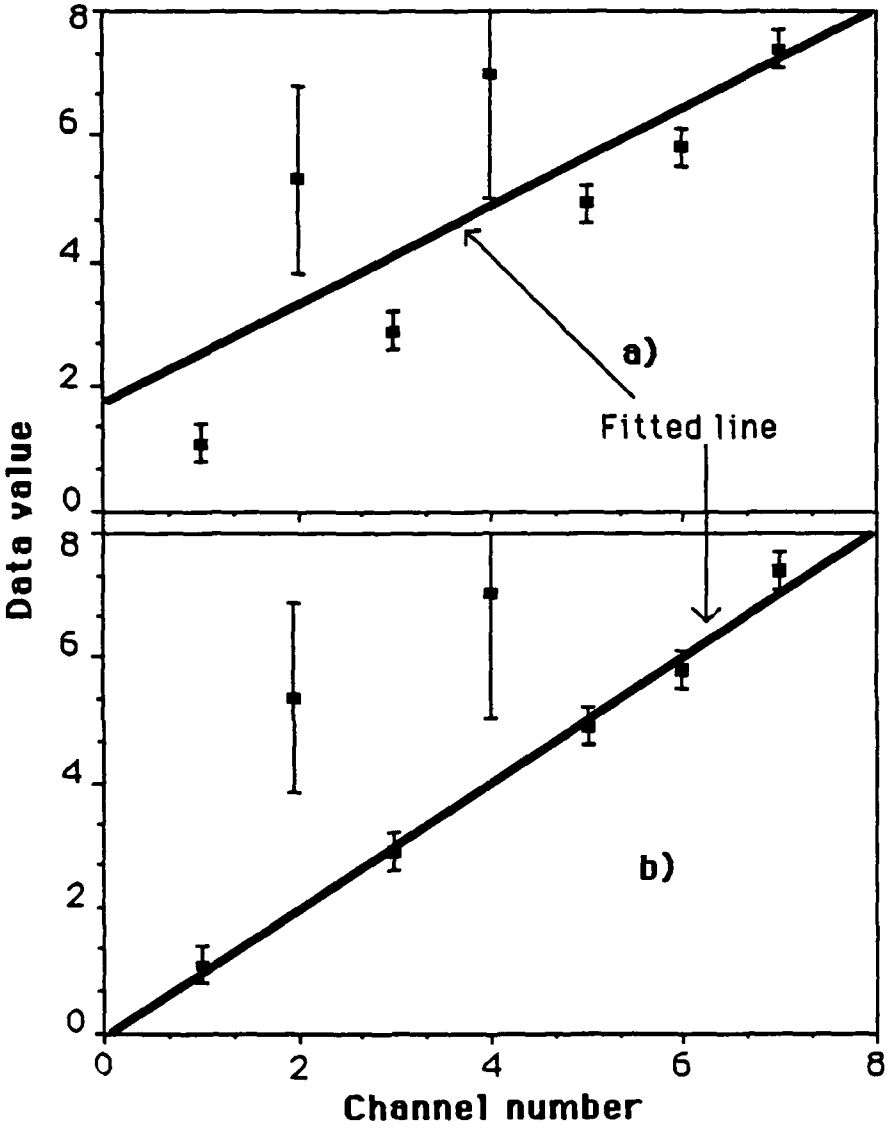


Figure 5.1 a) An unweighted fit to the data and b) A weighted fit to the same data.

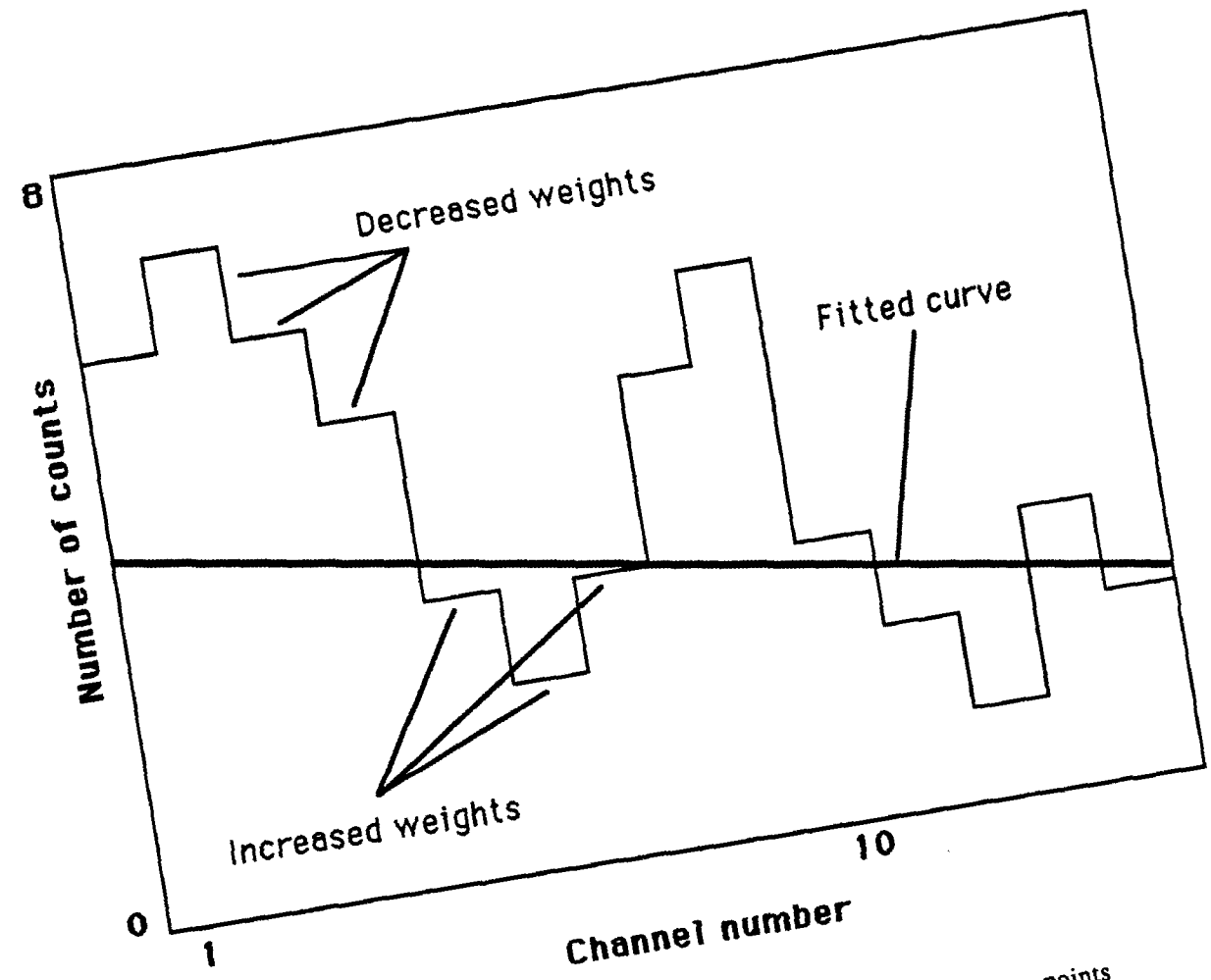


Figure 5.2 Illustration of the biasing towards low lying points

any fitting is done. The mechanism of choosing this option of the code is described further in the Appendix B users-guide.

Robust fitting is performed in ROBFIT by filtering the data over a 16 channel region. Before background fitting begins, the data are averaged in a robust manner, over successive 16 channel regions in which the weights of 'low lying' points are increased by a factor

$$A(1+\alpha(x-a)^2) \qquad \text{.....(5.1)}$$

and the weights of 'high lying' points are decreased by a factor

$$\frac{B}{(1+\alpha(x-a)^2)} \qquad \text{.....(5.2)}$$

where 'a' is the average over the region considered. A,B and α are adjusted so that the χ^2 minimum occurs when a=0 in the presence of one spurious point in every 16. These functions have been chosen to yield a χ^2 of 16 when summed over the 15 points that are distributed randomly in a Gaussian manner, together with the one spurious point. They also ensure that the average of the 16 point region is not disrupted by the spurious point. The method of determining A, B and α is given in Section 5.1.1.

The robust averaging has two advantages. The first, as described above, allows the background function to be estimated in the presence of peaks. The second is the increase in the operating speed of the code. All fitting to the background is now carried out on a reduced data set resulting in a speed increase of over 16 as compared with the raw data.

5.1.1 Determination of the robust fitting parameters

In this section we describe how the robust fitting parameters A,B, and α are determined during fitting. By using equations 5.1 and 5.2 as the weighting factors for points that lie respectively on the high and low sides of 'a', the average over a 16 point region, the code can be made insensitive to the presence of one spurious high-lying point in the 16 channel region. The averaging starts by choosing the 16 channel regions. Channel averaging starts at channel one and averages over the first 16 points, then moves on to channel 17 before averaging over the next 16 channel block. This procedure continues until the entire channel range has been averaged.

Within any 16 channel region χ^2 can be defined as

$$\chi^2=15\left(A\frac{\int_0^{\infty}(x-a)^2(1+\alpha(x-a)^2)e^{-\frac{x^2}{2}}dx}{\int_0^{\infty}e^{-\frac{x^2}{2}}}+B\frac{\int_0^{\infty}\frac{(x-a)^2e^{-\frac{x^2}{2}}dx}{(1+\alpha(x-a)^2)}}{\int_0^{\infty}e^{-\frac{x^2}{2}}}\right)$$

plus the contribution from the one spurious point

$$+B\frac{(\delta-a)^2}{(1+\alpha(\delta-a)^2)}$$

This minimizes at a=0 when

$$\left.\frac{\partial\chi^2}{\partial a}\right)_{a=0} = 0$$

$$= 15\sqrt{\frac{2}{\pi}}\left(-A\int_{-\infty}^{0}(2x+4\alpha x^3)e^{-\frac{x^2}{2}}dx + B\int_{0}^{\infty}\left[-\frac{2x}{(1+\alpha x^2)}+\frac{2x^3\alpha}{(1+\alpha x^2)^2}\right]e^{-\frac{x^2}{2}}dx \right)$$

$$+B\left[-\frac{2\delta}{(1+\alpha\delta^2)}+\frac{2\delta^3\alpha}{(1+\alpha\delta^2)^2}\right]$$

Changing variables to $z = \frac{x}{\sqrt{2}}$ in the first integral and evaluating the second gives

$$0 = 30\sqrt{\frac{2}{\pi}}\ A(1+2\alpha)$$ for the first term, and

$$-15\sqrt{\frac{2}{\pi}}\ 4B\int_{0}^{\infty}\frac{ze^{-z^2}}{(1+2\alpha z^2)}\left(1 - \frac{2\alpha z^2}{(1+2\alpha z^2)}\right)dz$$ for the second term, and

$$+B\left[-\frac{2\delta}{(1+\alpha\delta^2)}+\frac{2\delta^3\alpha}{(1+\alpha\delta^2)^2}\right]$$ for the third term.

The above equation will be solved by a Newton-Raphson technique to give α once A and B are known. A and B can be determined by requiring for A

$$1 = A\frac{\int_0^\infty x^2(1+\alpha x^2)e^{-\frac{x^2}{2}}dx}{\int_0^\infty e^{-\frac{x^2}{2}}} = A(1+3\alpha)$$

and likewise for B

$$1 = B\frac{\int_0^\infty \frac{x^2 e^{-\frac{x^2}{2}}dx}{(1+\alpha x^2)}}{\int_0^\infty e^{-\frac{x^2}{2}}} \quad \text{which gives} \quad B = \frac{1}{\sqrt{\frac{1}{\pi}}\ 4\int_0^\infty \frac{ze^{-z^2}}{(1+2\alpha z^2)}dz}$$

At this point A and B are still undetermined; however, by taking an initial value for α equal to $1x10^{-5}$, the calculated A and B can be used to predict a new value of α. Iterating on the A, B and α, calculations quickly finds the minimal values.

5.2 Representing the background with splines

A gamma-ray spectrum of 4096 channels can be expected to contain 200 or more peaks and to need about 100 coefficients to represent the background. Faced with such a complex situation, we must devise a method that will allow us to correctly describe each of these contributions. ROBFIT divides a spectrum into foreground and background. The foreground contains the peak contributions to the spectrum and is designated here by a function p(x).

During the fitting phase, constituent peaks in p(x) are each allowed to vary in overlapping groups in order to determine a new p(x). The background contains all the large structure contributions to the spectrum and is represented by a function $f_b(x)$. This background function is fitted as a sequence of moveable cubic splines. The power of ROBFIT lies in its ability to vary both the foreground and background functions in order to optimize the fit to the data. It iterates on background fitting, then foreground fitting. Following a background fit, a peak search is performed and new peaks are added. At the same time, all old peaks are re-calculated regarding their position, height and width. This iterative feature ensures that $f_b(x)$ remains a true representation of the background. The data, which consists of N measurements f_i at equally spaced points x_i, can be written in terms of the background and foreground as

$$f_i = f_b(x_i) + p(x_i) \pm \sqrt{\frac{1}{\omega_i}}$$

where ω_i is the channel weight assigned to the data value at channel i. Assuming x_i, f_i and ω_i are known, the first step is to determine the background function $f_b(x)$. If the code is started with no background and no peak information, it takes the background as a third order polynomial and p(x) as zero. It proceeds to fit this function to the data using the fitting methods outlined in Chapter 4. An improvement in both operation speed and accuracy can be effected by compressing the data by a factor of 16 before fitting, as described earlier in Section 5.1. This filters out high frequency components from the background and helps reduce any undetected peak contribution in the fit to $f_b(x)$.

Background fitting is accomplished by making $f_b(x)$ a weighted least-squares fit to a modification of the difference f_i-$p(x_i)$. The modified fi-p(xi) results from the fact that, until all the peaks have been found, this term contains peaks in addition to background. A schematic of how a fit is achieved is shown in Figure 5.3. The figure shows 10 channels containing an artificial peak with a width of 2 channels. Figure 5.3a illustrates the background fit with the peak contribution present. At this point, the background fit is simply a weighted average over the 10 channels. The peak region can be seen to stand out in the residuals shown on top of the figure. Figure 5.3b shows the improvements in the background determination once the peak contribution has been removed.

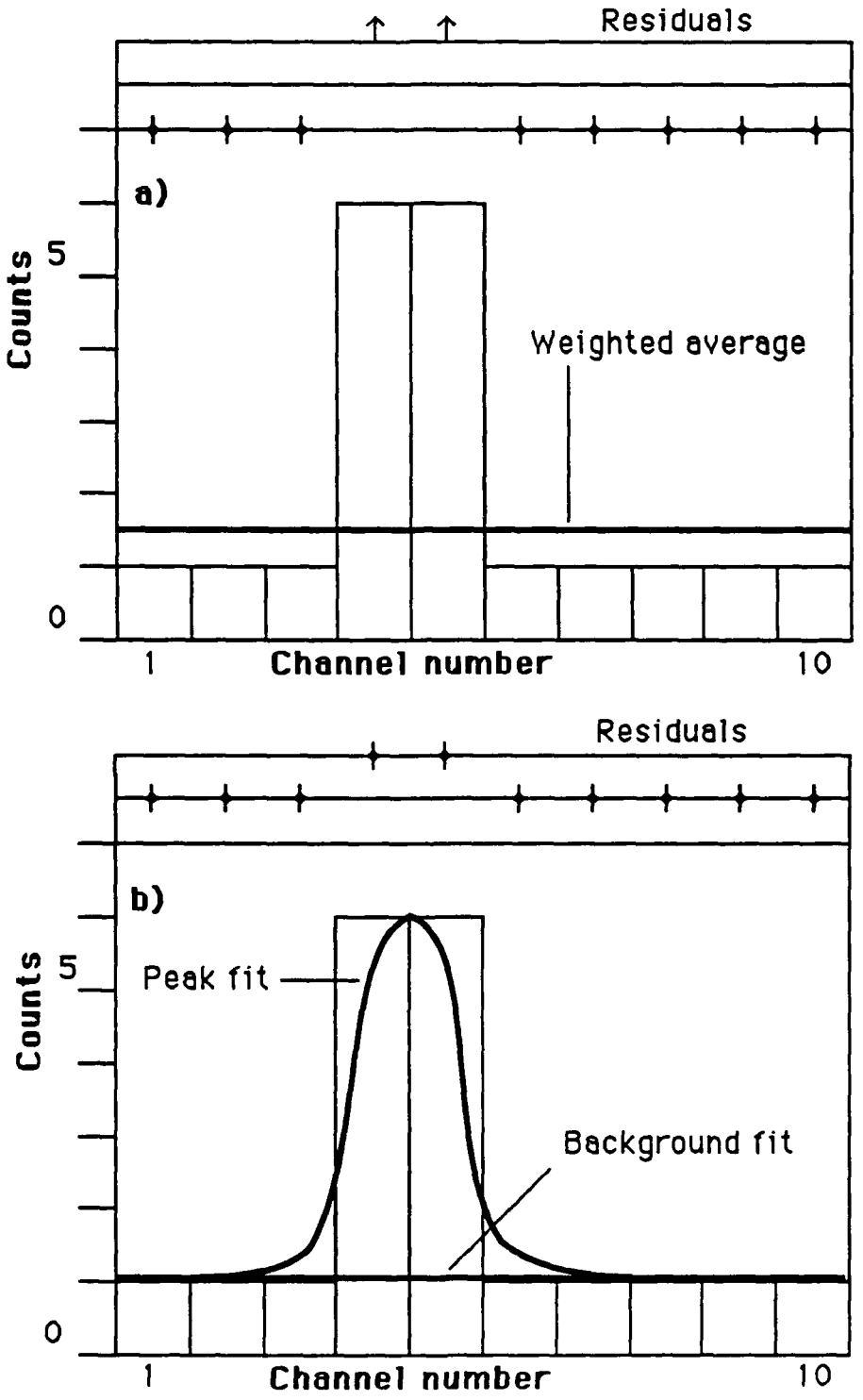


Figure 5.3 a) A background fit to a 10 channel region containing a peak. b) The improvement in the background fit once the peak has been found. Note that the peak has dropped with the background and would be raised again in the next iteration.

After the first round of peak fitting, which is described further in Chapter 6, f_b is redetermined. At this stage, knots are added to the background. Two knots are added at each background refit until the user-specified maximum number has been reached. Re-optimizing the background and peak parameters is an important feature of ROBFIT as it correctly accounts for the correlations between them. Using splines ensures that f_b is a smooth, continuous function, with a structure that has been determined using the entire data stream.

Within the startup sequence of the code, the user chooses the number of knots to add to the background. The number of knots chosen must be consistent with the quality of the data. After fitting, the user must view the fit and decide whether the background has been correctly determined. A 'better' background fit can always be achieved by adding more constants. In curve fitting of this nature, there is a trade off between background and small peak detectability. If, for example, a large number of knots are used in the background fit, the f_b will tend to 'follow' rises and dips within the data. Consequently, small peaks may be washed out. The technique we use to get around this problem is to start with a 'stiff' background containing only a few knots and progressively 'slacken' it until the required flexibility has been achieved. This point can usually be defined by the size of the added background splines; a list of which is kept in the background constants file (see Appendix B). Once the optimum number has been reached, additional splines do not contribute significantly to the fit. This can be seen by simply inspecting the constants file and monitoring the size of the constants just added.

This background fitting technique is similar to the methods described by Morton[21]. The position of the knots is not so critical, because the code will minimize χ^2 with respect to the knot positions. A bad knot placement will be forgiven by the minimizing routine as ROBFIT will move it to the best position during the χ^2 reduction. ROBFIT adds background knots, two at a time, so that they can follow the natural rise and fall of the background. It has been found to be satisfactory to have the code initially place the first new knot at the middle of the region with the largest 'robust' error, and the second a reasonable distance above it. Knot positioning is described in detail in Section 5.3. The minimization routine SMSQ then moves these and all other knots to their optimum positions.

ROBFIT contains options for selecting various background fitting scenarios. If a previous background fit has been done, the set of background coefficients determined in that run can be fed into the new fit. There is no need to recalculate the background because the code has been supplied with a starting point for fitting. Additional constants can be added to this starting file if a more detailed fit is called for. This is carried out by selecting the *'continue adding knots'* option. Other operations may need no background fitting, or fitting of a fixed background function. Choosing the *'no new knot'* option tells the code to hold the background fixed at the starting configuration; while the *'no background fit'* option performs only a peak analysis.

When comparing similar spectra, it may not be desirable to adjust the position of the background knots from one fit to the next. This can be accomplished by fitting successive spectra with the spline constants fixed at positions determined in a previous fit, and variable only in size. This cuts the number of moveable coefficients in half and preserves the quality of the background from one fit to the next. It also allows errors in the differences between spectra to be estimated with fewer degrees of freedom. The *'FIXK'* option in ROBFIT allows for this kind of fitting. As a bonus it is also much faster than the normal fitting with variable knots. The above options are explained further in Appendix B.

5.2.1 Linear and exponential fitting of the background

ROBFIT has two modes for fitting the background; exponential and linear. In an exponential fit, the background is expected to vary as an exponential function. Selecting an exponential background fit helps the fitting function quickly rise at lower channel numbers; a common feature of many background functions, and in doing so, better represents this type of background. In linear fitting, no transformation is used and a straight-forward linear fit is performed on the spectrum.

The background is positive definite in most spectra. Choosing an exponential fit makes use of this information and helps control the shape of the background. It must be remembered, however, that you can not fit negative backgrounds with the exponential option switched on. With subtracted spectra, or spectra in which the background does go negative, a linear fit must be carried out. In linear fitting, the background is expected to be a relatively smooth function that can be fitted with a few constants. We

recommend the use of the exponential fitting function for most spectra, even though this fit takes slightly longer than the linear case.

5.3 How ROBFIT determines where to place the knots

To summarize, the general technique used in fitting the background is to successively increase the number of knots in the representation of the background function. ROBFIT must, therefore, have some mechanism for choosing where to place the extra knots. The actual positioning is not critical as the minimization routine can move the knots to their optimum positions. However, a bad positioning can take a considerable amount of computing time to correct. It is better to give the code a good starting point, thus limiting the range over which the minimization routine has to vary the knots.

The knots are actually added two at a time so that rises or dips in the background can be closely followed. One spline follows the upward trend while the other follows the downward trend. Adding knots two at a time means that only small changes in the residuals occur. This means that the minimization procedure, which relies on small variations from one fit to the next, can operate correctly.

The first knot is positioned at the largest residual in the compressed data. The second knot is placed a number of channels above this position. The channel offset being chosen as the maximum of five, or $\frac{NN}{2*NV}$, where NN is the number of compressed channels and NV is the number of knots already present. This has the effect of selecting a smaller fitting region as the number of knots increases. This helps fitting speed and accuracy. A review of knot positioning in least-squares fitting using cubic-splines is given by Morton[21].

6 Representation of peaks.

In this chapter we show how ROBFIT handles the peak component of a spectrum. We describe how the peak regions are located and fitted with a given peak shape. We also show how the code decomposes multiple peak regions and how peak width, area, and position parameters are varied to optimize the separation of background and foreground functions.

6.1 Spline representation of peaks

You may expect to find a variety of peak shapes within a spectrum. A standard must be created for each of these shapes before running ROBFIT. Each standard is used by ROBFIT to determine where peaks of that shape reside. The mechanics of creating a standard are explained in Appendix B. Here we concentrate on the mathematical representation of these standards. The standard peak is expected to be a relatively small set of N data values f_i, with weights ω_i at positions x_i. The user can create a standard either from a peak within the data or choose one of the peak generating functions supplied with the code (Lorentzian, Gaussian, Voigt). In the case of a fit to a peak within the data, the chosen peak is expected to be well separated from other spectral features. A fit to the peak region can then be made by assuming only a simple underlying background function.

The fitting function contains a user-specified number of polynomial coefficients to represent the background, and a user-specified number of back-to-back cubic splines to represent the peak. The back-to-back cubic splines are given by

$$\Gamma_S(x) = ((1+x)_+(1-x)_+)^3$$

where

$$(x)_+ = 0 \quad \text{for} \quad x \leq 0$$
$$(x)_+ = x \quad \text{for} \quad x > 0$$

Figure 6.1 shows the typical shape of a back-to-back cubic spline. The splines are ideal for representing peaks because they have components that naturally follow the rise and fall of the peak shape.

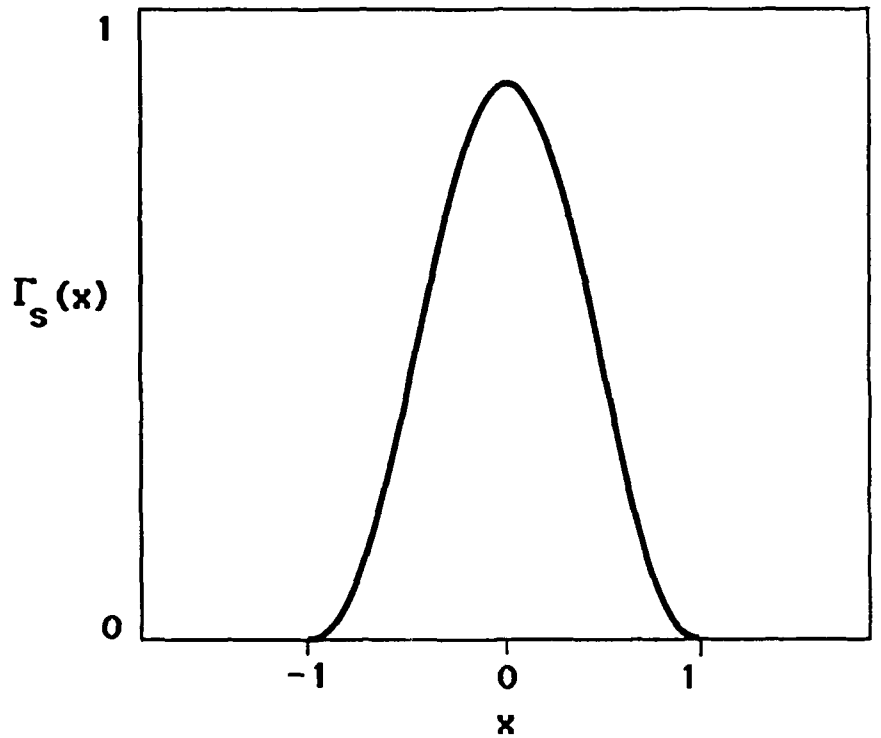


Figure 6.1 Illustration of the shape of a back-to-back cubic spline.

Specifically, the N data points f_i are fitted to the form

$$g(x) = BG + PK$$

where BG is the background contribution

$$BG = \sum_{k=1}^{K} a_k x^k$$

with K as the user specified polynomial coefficient. Since this fit is over a limited region selected by the user, the background variation is not expected to be significant. This makes the polynomial representation adequate. When generating one of the standards supplied with ROBFIT, K is set to zero because, in this case, we have a peak with no background.

PK is the peak contribution

$$PK = \sum_{l=1}^{L} b'_l{}^2 \Gamma_s\left(\frac{(x-p'_l)}{w'_l}\right)$$

with L as the user specified number of back-to-back cubic splines, p'_l and w'_l are the starting values of the positions and widths of the splines. These are calculated by the code so that

$$g(x) = \sum_k a_k x^k + \sum_l b'_l{}^2 \Gamma_s\left(\frac{(x-p'_l)}{w'_l}\right)$$

The optimum fit is then calculated by minimizing χ^2

$$\chi^2 = \sum_i \omega_i (f_i - g(x_i))^2$$

with respect to a_k, b'_l, p'_l and w'_l. In practice, an additional term is added so that the fitted peak width is kept larger than four channels. This gives

$$\chi^2 = \sum_i \omega_i (f_i - g(x_i))^2 + \sum_i (4 - w'_i)_+^3$$

Non-linear minimization is carried out using the method described in Chapter 4. Starting with a single Γ_s of initial width $w'_1 = 2$, and position $p'_1 = x_k$, where f_k is the largest data point and $b'_1{}^2$ is equal to the difference between f_k and the curve fit at that channel. At the outset of fitting, the curve fit is a linear interpolation between the end points of the selected region. This starting configuration is illustrated in Figure 6.2a. Once the optimum values of a_i, b'_l, p'_l and w'_l have been found, the contribution from this spline can be removed and the residuals searched again. A second Γ_s is added with $w'_2 = 2$, $p'_2 = x_k$, the position of the largest residual, and $b'_2{}^2$ equal to the residual at the k^{th} channel. See Figure 6.2b.

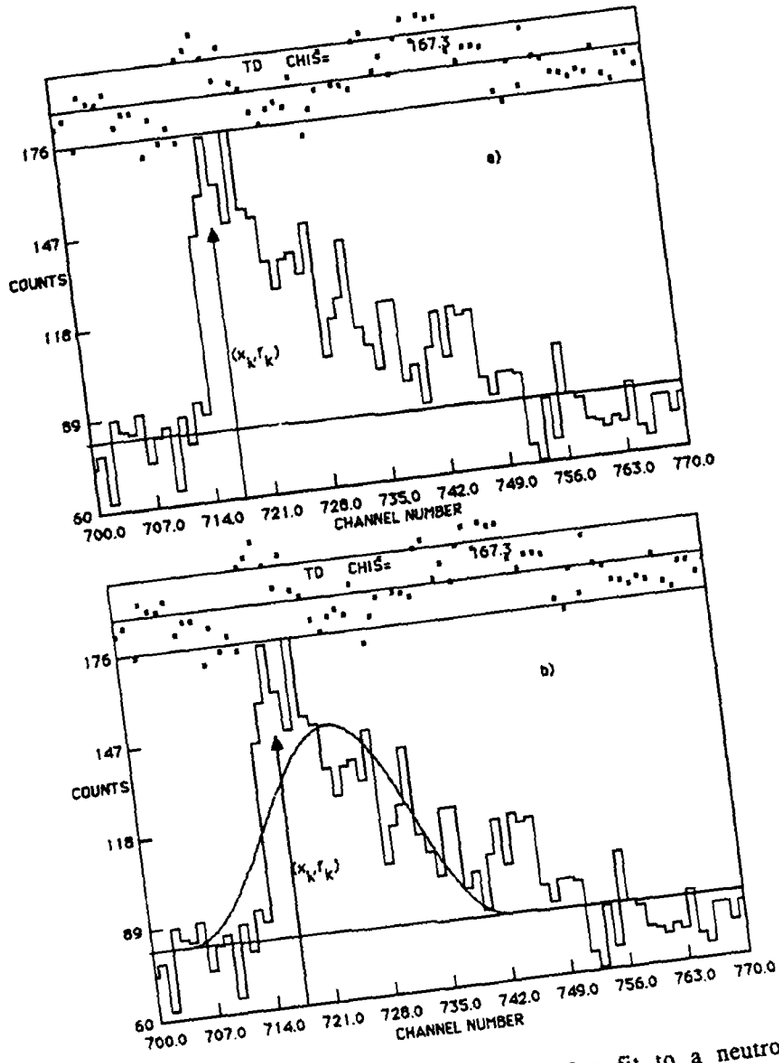


Figure 6.2. (a) The starting point of a fit to a neutron peak region of the Supernova data, (b) the first back-to-back cubic spline is added. See Figure 6.5 for the situation after a second back-to-back spline is added.

The function χ^2 is again minimized with respect to both old and new parameters, and the entire process repeated until the appropriate number of splines have been included.

The splines are then normalized so that

$$\Gamma(x) = \sum_{l} B_l^2 \Gamma_s\left(\frac{(x-P_l)}{W_l}\right) \qquad \text{.....(6.1)}$$

represents the standard peak. B_l^2, P_l, and W_l have been scaled to make the height and width of the peak equal to unity.

The use of the positive definite $\Gamma_s(x)$, makes the form chosen for fitting the standard peak positive definite. This provides a rather large amount of physical intuition about the nature of the spectrum being fitted. This helps clarify what should be the background, and what should be the peak. It will be a nuisance if one is trying to generate a standard for some purpose which requires a negative portion.

The normalized standards will be used when fitting the spectrum. In its peak fitting phase, ROBFIT utilizes a parameterized function,

$$p(x) = \sum_{l} b_l^2 \Gamma\left(\frac{(x-p_l)}{w_l}\right) \qquad \text{.....(6.2)}$$

whose component parts are the standard peaks of equation 6.1, to represent the peaks. This enables the code to re-scale the standard to any height, or width, or position it at any location. Note that p(x) is also positive definite.

6.1.1 Gaussian and Lorentzian standards

Both Gaussian and Lorentzian standards can be generated automatically using ROBFIT. The shape is chosen by selecting either *'GAUS'* or *'LORE'* on the *'standard-generating'* menu pages described in Appendix B. When fitting a pure peak shape, for example a Gaussian, there is no background under the peak and the number of background coefficients must be set to zero before generation begins.

Gaussian is generated by using

$$\text{Gaus} = \frac{1000}{\sigma\sqrt{2\pi}} \exp\left[-\frac{1}{2}\left(\frac{(x-\mu)^2}{\sigma}\right)\right]$$

which gives a peak height of 1000. The peak is positioned at channel zero (μ=0) and its width is set to a FWHM = 2.

Figure 6.3 illustrates how the splines can be used to increase the accuracy of the fit. The residuals between the spline curve fit and the data values are shown above each figure. The center line is the zero residual level, while the top and bottom lines are the $+2\sigma$ and -2σ levels respectively, where σ is calculated from $\sqrt{\text{Max data value.}}$ Figure 6.3a is a fit to a Gaussian using three back-to-back cubic splines. It shows that there is still structure in the residuals, indicating that the fit could be improved. By the time seven back-to-back splines have been added, the accuracy of the fit has been significantly improved (Figure 6.3b).

In order to detect small peaks within the data, all structure from the standard must be reduced to a level that is small compared to the peak size. For example, if the standard in Figure 6.3a is used to fit a peak in a full spectral fit, ROBFIT will attempt to fit the residual fluctuations caused by the mismatch between the standard and the peak shapes, with additional small peaks. This leads to a situation in which each peak of the spectrum is decomposed into many secondary peaks; this effect is called braiding. Once the Gaussian has been fitted, the splines are normalized to give heights and widths unity as described in Section 6.1.

Lorentzian standards can be generated in a similar manner. These are created by a function,

$$\text{Lore} = \frac{1000}{\pi} \frac{\text{FWHM}/2}{(x-\mu)^2+(\text{FWHM}/2)^2}$$

which gives a peak height of 1000 counts. The FWHM is set to 2 channels and is positioned at the origin (μ=0).

The user may wish to generate a peak shape that can be expressed analytically, but is not one of the standard shapes. It is easy to change the *'shape-generating'* function by simply including a FUNCTION which calculates the required shape in SUBROUTINE BLI in the VSHAPE routine. Both VSHAPE and BLI are shown in full in Appendix A.

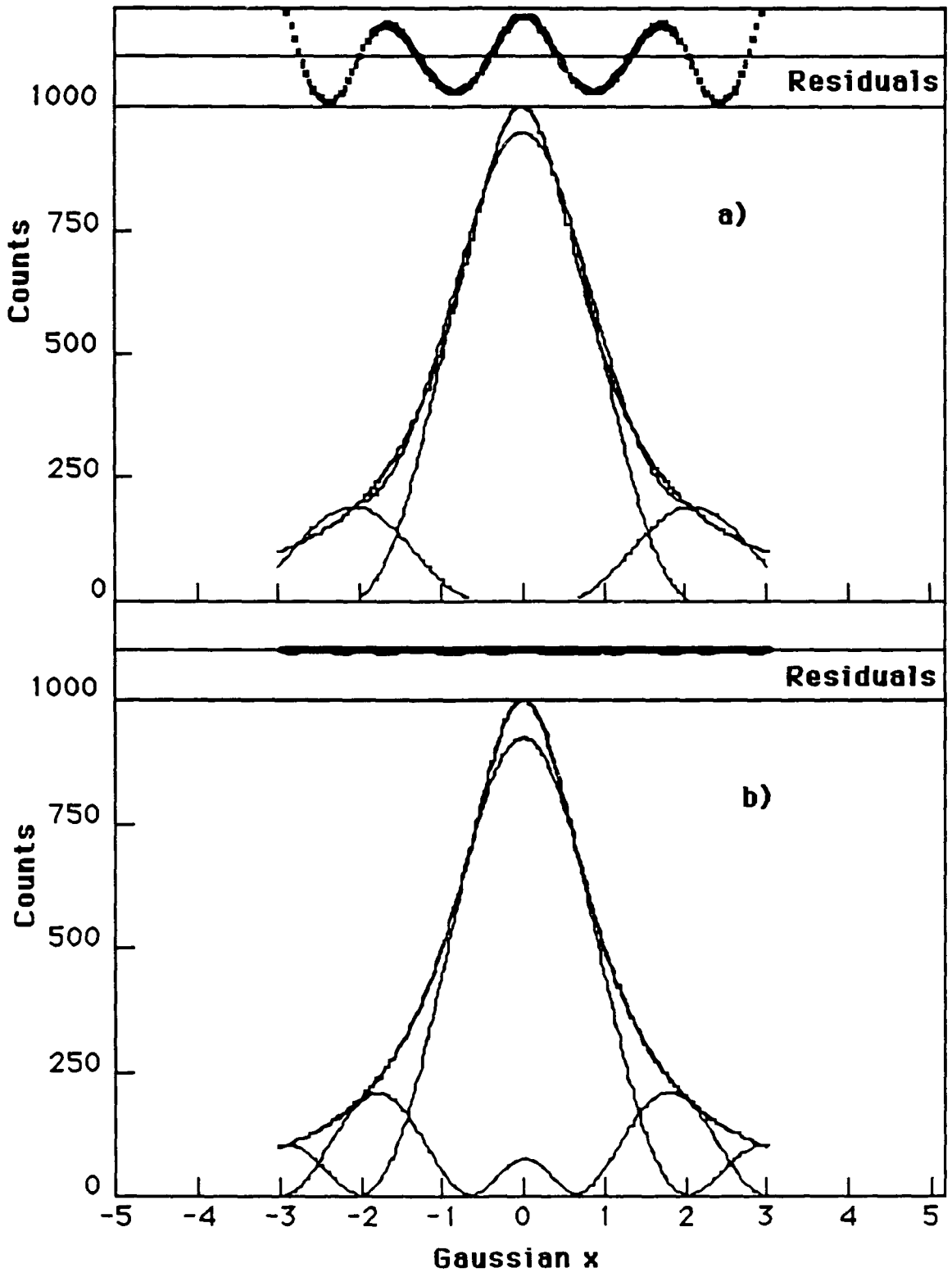


Figure 6.3 a) A three spline fit to a Gaussian and b) the increase in accuracy using seven splines.

6.1.2 Voigt standards

Another useful peak shape to have on hand is the Voigt. This shape is composed of both Gaussian and Lorentzian components. For example, the intensity distribution in a spectral line broadened by two independent effects is expressed by the equation

$$f(x) = \int_{-\infty}^{\infty} G(y)L(x-y)dy$$

where G and L are the line profiles that would result if the individual broadening functions were present alone. In most cases, G(x) is a Gaussian distribution and L(x) is a Lorentzian distribution.

One prescription used by Finn and Muggestone[22] helps to illustrate the η mixing parameter that ROBFIT uses to generate Voigt profiles.

$$f(x) = \frac{a}{\pi} \int_{-\infty}^{\infty} \frac{e^{-y^2}}{(x-y)^2+a^2}\, dy$$

where

$$a = \frac{\Gamma_L}{4\pi\Gamma_G} = \frac{1}{\eta}$$

with Γ_L = full width at half maximum of the Lorentzian and Γ_G = full width at half maximum of the Gaussian.

The η parameter effectively determines the mix between Gaussian ($\eta << 1$) and Lorentzian ($\eta >> 1$). The Voigt profile is normalized to a height and full width at half maximum of one. All the user has to do to generate a Voigt profile is to define the η mixing parameter. Figure 6.4 illustrates the variation of η over a range η = 0.1, Lorentzian; to η = 10, Gaussian.

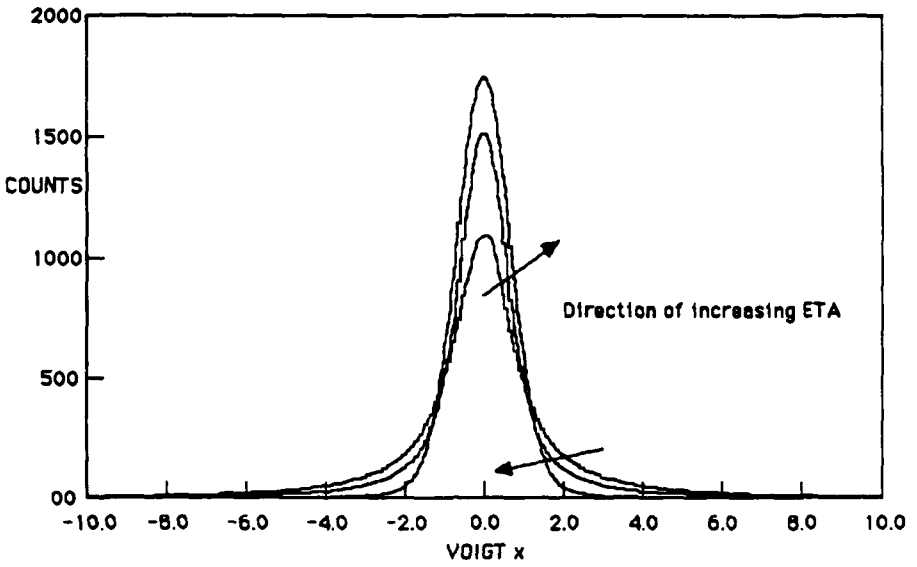


Figure 6.4 Variation in peak shape with changes in the VOIGT ETA parameter.

6.1.3 Selecting a standard from the data

Certain spectra contain peak shapes that are neither Gaussian, Lorentzian, nor Voigt. The spectrum in Figure 6.5 is taken from the supernova data and illustrates how a neutron inelastic scattering standard can be generated. Chapter 8 describes how this standard is used in fitting the data. The problem with the neutron peaks is that their shapes are not well defined. Therefore, we must generate this standard from the raw data. The peak indicated in the figure can be seen to be well separated from other spectral features and resides on a simple background. Under these conditions we can create a standard using the procedures detailed in Appendix B. A region surrounding the peak is chosen and fitted using a polynomial fit to the background, and the back-to-back cubic spline fit to the peak. Figure 6.5 illustrates the result of a 2 constant fit to the background and a peak fitted with 2 back-to-back cubic splines. The figure shows little fluctuation in the residuals, indicating that the peak has been fitted well.

6.1.4 Selecting multiple peak shapes

When a real spectrum is fitted, usually no one standard will represent all peak shapes. Consequently, ROBFIT has the capability of using up to five standards in any one fit. At runtime, the code decides which of the peak shapes

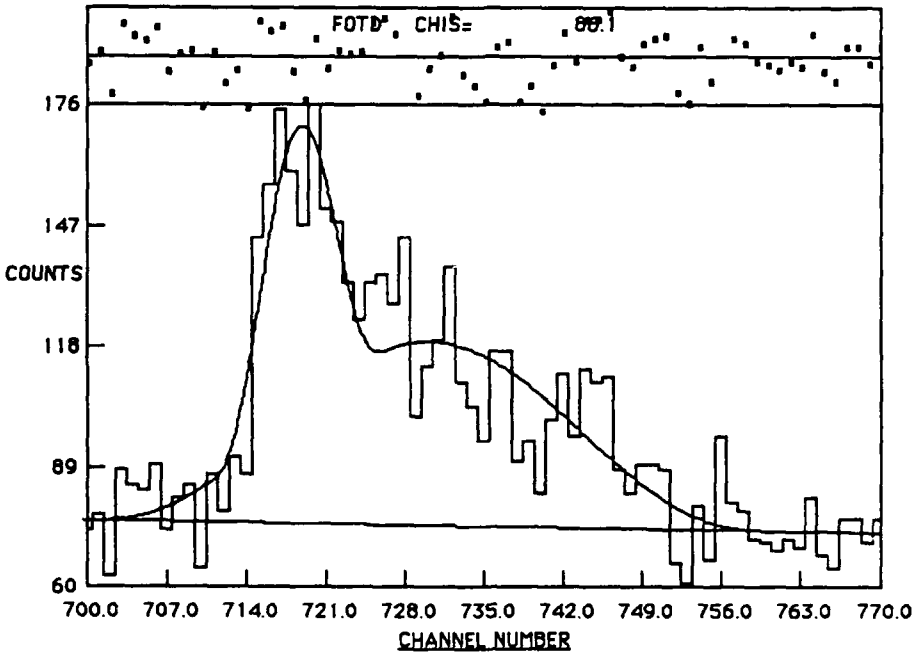


Figure 6.5 The supernova neutron peak region fitted with a 2 constant background and 2 back-to-back cubic spline fit.

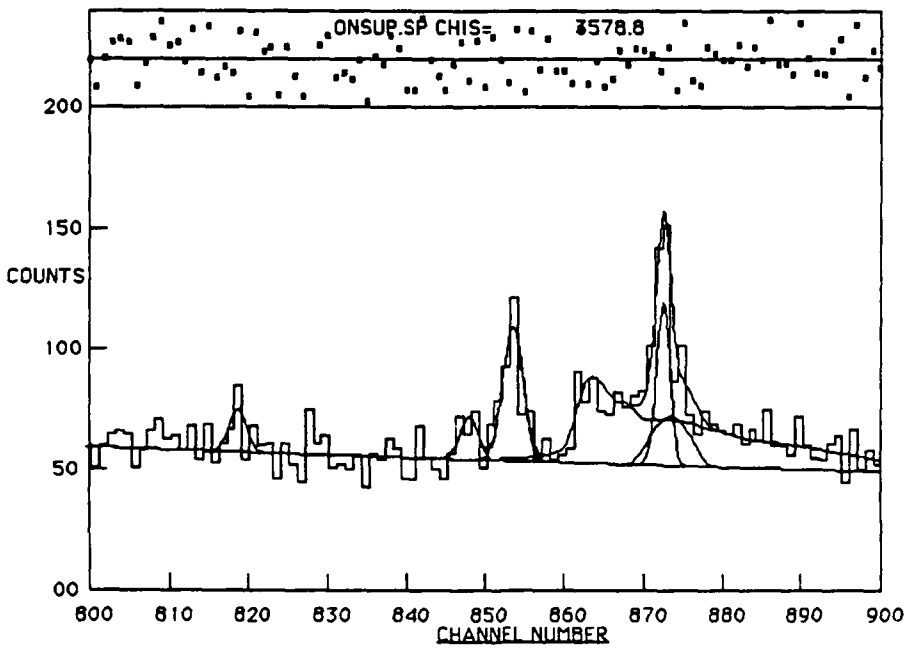


Figure 6.6 A fit to the neutron peak region of the supernova spectrum.

is more appropriate to fit a given region of the spectrum. This selection procedure is described further in Section 6.3.

Each of the peak shapes to be used in the fit is generated as described in the previous sections. These standards files are then read into the full spectral fit at the start of the fitting sequence. ROBFIT will fit individual peaks and decompose overlapping peak regions, taking into account all shapes present in the fit. Figure 6.6 shows a fit to the neutron peak region of the supernova spectrum using three different standards.

6.2 Initial singlet detection

Before getting into the mathematics of peak detection, it is useful to describe the flow of the peak finding process. Central to this process is the setting of a CUTOFF level. This level is user selectable and defines when the code will stop searching for peaks. The code will find peaks until there are no residuals greater than this CUTOFF level. The residual for a potential peak at x_i is the excess counts in the data in a region equal in size to the full width at half maximum of the potential peak dividend by the error in the integral of the fit across this same region. Having determined that there are residuals greater than the CUTOFF, the code proceeds to find the channel position of the largest residual. This is the channel at which the code will attempt to put the peak. A small region surrounding the largest residual is then selected. The χ^2 over this region is then minimized with respect to the peak parameters. At this point the background has not been re-calculated. Due to the addition of this peak, the background may have been disturbed within a region surrounding the peak. Consequently, on the search for the next largest residual, the code does not position peaks within this region. Both the background and the peak constants are re-optimized following peak fitting. This ensures that both background and peak functions correctly describe the data and are not biased by fitting the peaks in small regions. Figure 6.7 illustrates a flow diagram for the peak fitting sequence.

The code has been designed so that a number of successively smaller CUTOFF runs can be executed consecutively. ROBFIT produces the fitted parameter file after each CUTOFF run. This stage by stage reduction allows the user to step through the fitting sequence and monitor the progress of the fit.

We are now ready to begin the mathematics of peak fitting. To recap, the

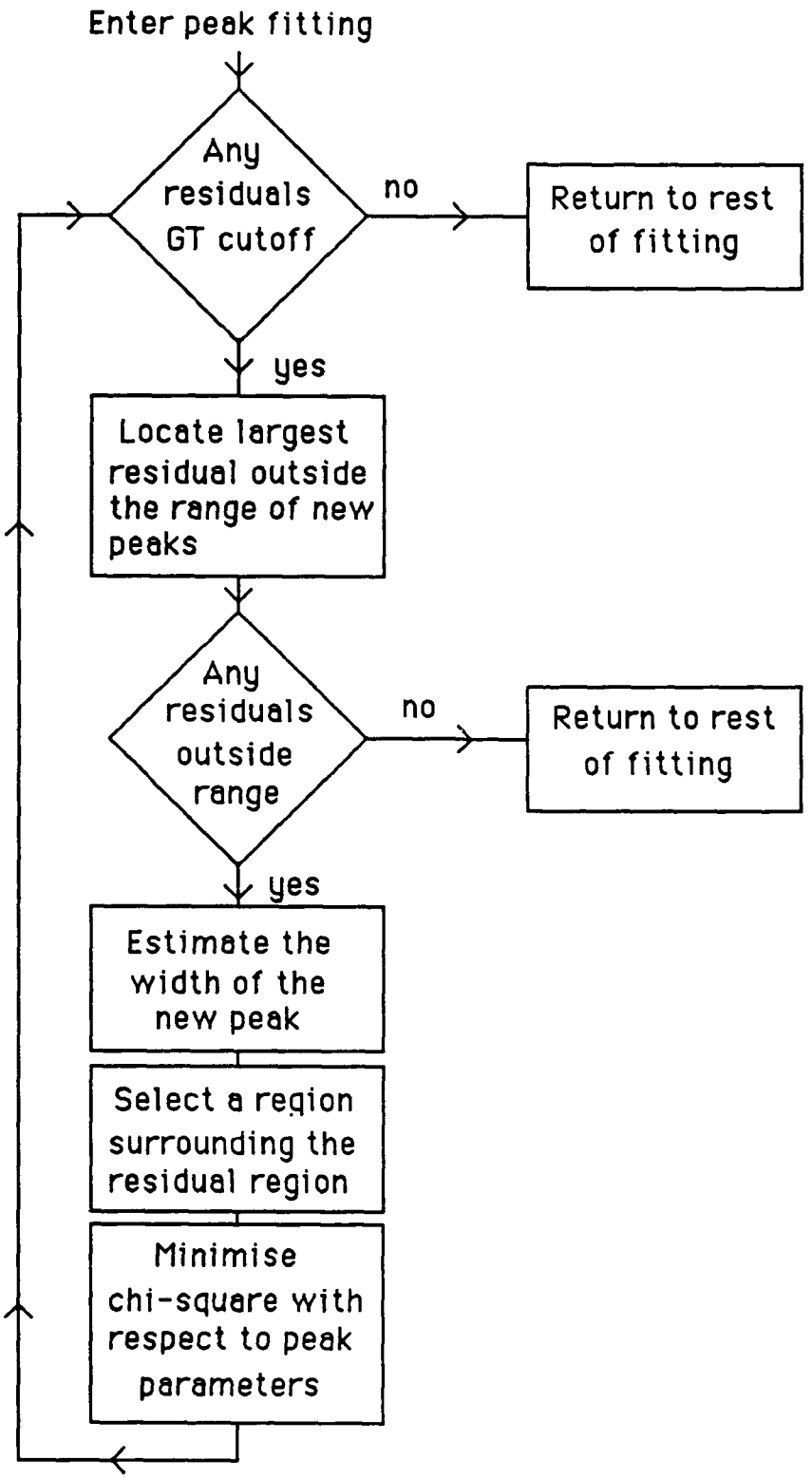


Figure 6.7 Flowchart of the peak-fitting sequence.

fitting function representation begins with N measurements f_i at equally spaced points x_i. It is assumed that we have found a relatively slow varying function $f_b(x)$ so that when it is added to the function p(x) representing the peaks, the measurements are given by

$$f_i = f_b(x_i) + p(x_i) +/- \frac{1}{\sqrt{\omega_i}}$$

where all quantities have been defined in Section 5.2. Composite back-to-back cubic spline standards are assumed to represent the peaks. All different peak shapes within the spectrum are given a different standard. These standards have been described in Section 6.1.

Each standard is normalized to make the standard peak have height and full width at half maximum unity. Equation 6.2 defines the parameterized peak function p(x). Two important features of the composite splines, $\Gamma(x)$, are that they are identically zero for most values of x and their derivatives can be found exactly.

The following outlines the sequence of events involved in peak location. In order to locate peaks, a set of smoothed residuals,

$$R_i^2 = \frac{1}{FWHM} \sum_{j=i-2}^{j=i+2} \omega_i (f_i - f_{th}(j))^2 \qquad \text{.....(6.3)}$$

is searched for the largest residual R_L, where f_{th} is the theoretical estimate of the fitting function, and is equal to the sum of the background and peak functions. FWHM is the full width at half maximum determined from the user-supplied width range, and ω_i is the weight at channel i. The value x_1 of the first peak standard, given by equation 6.2, is then initialized to x_L, the position of the largest residual. The width of the standard w_1 is set to an interpolated value between the user-supplied high channel and low channel estimates, which are read into ROBFIT at the outset of fitting. A minimization using the method described in Chapter 4, is then carried out on the quantity

$$\chi^2 = \sum_j \omega_j (f_j - p(x_j) - f_b(x_j))^2$$

with respect to the parameters in p(x). For a single standard this would be a minimization of the position x_1, width w_1, and scaling factor b_1 of the standard. The optimization is performed over a limited region of the spectrum such that

$$x_1 - 3w_1 \leq x_j \leq x_1 + 2w_1$$

The limited range of x_j reduces the computational time while the short range nature of the peak keeps it from sacrificing accuracy. After minimization, the matrix of second derivatives of χ^2 is inverted, without smoothing coefficients, to provide error estimates for the peak parameters. The effect of this peak can now be added to our theoretical representation of the function f_{th}.

The residuals are now recalculated, and large spectra can be searched again for other peaks. The fact that the peak just found was present in the determination of the background function f_b, indicates that the background will be inaccurate in the vicinity of this peak. Thus, the code is not allowed to introduce another peak in this region until the background has been re-calculated. Peaks are not added within a channel range $6w_p$ on either side of the new peak, where w_p is the width of the peak just added.

When there are no more well separated residuals greater than the user-supplied CUTOFF, the background is recalculated. Following the background refit, all channels are again made eligible for largest residual status. Additional peaks can come from two sources:

a) The residual was in the range of another peak on a previous peak-fitting iteration, and

b) The background has been lowered significantly, allowing the residual locater to find a new peak.

If there are peaks within range of this new residual, where again the range is determined by the user-supplied width values, ROBFIT first re-minimizes the χ^2 with respect to the old peak parameters. This is done because the background may have changed significantly since the old peaks were first added. After the re-minimization, if the χ^2 does not change significantly, the new peak is added and χ^2 minimized with respect to the parameters of all peaks. This process is true for any number of peaks, up to the maximum of 10

within any one region. This sequence of events is shown in Figure 6.8.

Once ROBFIT determines that there are no residuals above the CUTOFF, the code refits all peaks one last time before ending that particular CUTOFF run.

An important part of ROBFIT is allowing peak positions, widths, and strengths to vary. These variations allow the code to explore the "best fit" parameter space and thus estimate the errors on the parameters more accurately. There are three ways in which ROBFIT can manipulate the peak parameters.

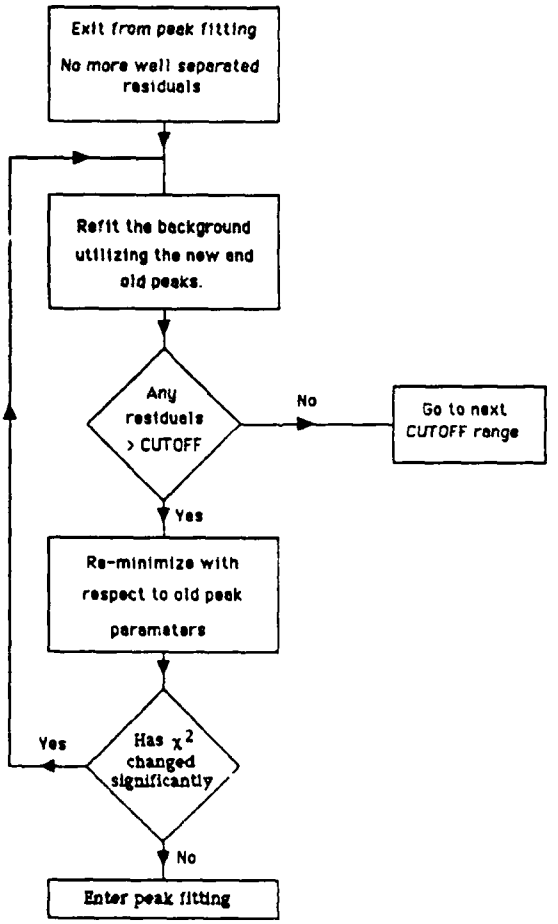


Figure 6.8 Flowchart of the peak-adding sequence

The first method is variation during fitting. This has been discussed in the above peak parameter evaluation. ROBFIT continually fits and refits all

peak parameters during its cycling so that all constants and their associated errors have been optimally determined.

The second method is the case when the peak widths are known. ROBFIT contains a mode of operation in which the peak widths are held constant. Running the code in this manner greatly speeds up its operation as the number of fitted constants drops by one third.

The third and final method is that of modifying individual peak parameters prior to fitting. ROBFIT has the ability to read in peak constants from a previous fit. This gives the user the choice of modifying the input file. Peak constants can be changed, or new peaks added, or old peaks deleted, as required. During fitting, added peaks that do not significantly reduce χ^2 will be removed, or if modified parameters are outside the limits of the "best fit", they will be re-calculated. This provides a powerful tool when analyzing spectra. The user can now experiment with the contributions to the peak list and see the effects on the quality of the fit.

6.3 Dealing with spectra with more than one peak

Section 6.2 outlines the mechanics of locating and fitting peaks; however, only a single standard was considered. The situation is further complicated when the fit has to be performed with more than one standard. Now the residuals locator of equation 6.3 must be run for each standard, and the largest residual from all standards chosen as the peak position. In order to select the correct standard for this portion of the spectrum, ROBFIT performs a fit to this peak position for each standard. Only the parameters of the standard being fitted are allowed to vary. All peaks already present fixed within explicitly defined ranges. This frequently allows otherwise unresolvable multiplets to be broken by giving ROBFIT extra information to use.

6.4 Fits in which only peak heights are allowed to vary

In some situations, for example in analyzing repeated sections of a spectrum taken with differing statistics, it is useful to be able to hold the positions of the peaks and the shape of the background constant in order to study the variation in peak heights from spectrum to spectrum. The user can accomplish this by simply selecting the *'FIXK'* option from the ROBFIT menu (see Appendix B).

7 Treatment of errors

One of the most important features of ROBFIT is its ability to accurately determine the errors on fitted quantities. In this chapter we discuss the error calculations that are used, and show how they are implemented in the code. We begin by giving a general introduction to error analysis, and then we extend this to the determination of the errors in the least-squares fitting constants. In the last three sections we describe the calculations involved in determining the errors in the parameters of a fit.

7.1 General error analysis

Assume that there exists a function q of the data points which yields the exact result when applied to a set of data points containing no random error. When this function is used with N data points, f_i, each containing differences Δf_i from their exact values, q_{ex}, then

$$q(f_1,f_2,.....f_N) = q_{ex} + \sum_{i=1}^{N} \frac{\partial q}{\partial f_i}\Delta f_i \qquad .....(7.1)$$

While Δf_i is difficult to ascertain, in general, an ensemble of different, but equivalent, data points is expected to yield

$$\begin{aligned} \{\Delta f_i\} &= 0 \\ \{\Delta f_i \Delta f_j\} &= 0 \\ \{(\Delta f_i)^2\} &= \varepsilon_i^2 \end{aligned} \qquad .....(7.2)$$

where, in general, ε_i^2 is the expected variance in the i^{th} data point. Thus

$$\{(q(f_1...f_N)-q_{ex})^2\} = \sum_{i=1}^{N}\frac{\partial q}{\partial f_i}\sum_{j=1}^{N}\frac{\partial q}{\partial f_j}\{\Delta f_i \Delta f_j\}$$

$$= \sum_{i=1}^{N} \left(\frac{\partial q}{\partial f_i}\right)^2 \varepsilon_i^2 \qquad \text{.....(7.3)}$$

which reduces the error problem to that of finding the dependence of the calculated result on the input data.

In the following sections, equation 7.3 will be examined in detail for its applications in ROBFIT, but it should be mentioned that errors can be calculated with equation 7.3 as it stands. This is done by physically changing all data points and recalculating the q of interest. This yields

$$\frac{\partial q}{\partial f_i} = \frac{q(f_1,....f_i+\varepsilon,f_N)-q_0}{\varepsilon} \qquad \text{.....(7.4)}$$

which, along with ε_i, is all that is needed for equation 7.3.

7.2 Coefficients determined by least-squares fitting

The coefficients c_k are determined by minimizing

$$\chi^2 = \sum_i \omega_i (f_i - f_{th}(c_n, x_i))^2$$

$$= \sum_i \left(\frac{f_i - f_{th}(c_n, x_i)}{\varepsilon_i}\right)^2 \qquad \text{.....(7.5)}$$

with respect to c_j. The first derivatives of χ^2 with respect to c_k are

$$\frac{\partial \chi_0^2}{\partial c_k} = -2 \sum_i \omega_i (f_i - f_{th}(c_n, x_i)) \frac{\partial f_{th}(c_n, x_i)}{\partial c_k} \qquad \text{.....(7.6)}$$

and the second derivatives are

$$\frac{\partial^2 \chi_0^2}{\partial c_l \partial c_k} = 2 \sum_i \omega_i \frac{\partial f_{th}(c_n, x_i)}{\partial c_l} \frac{\partial f_{th}(c_n, x_i)}{\partial c_k} \qquad \text{.....(7.7)}$$

in which terms of order

$$\frac{(f_i - f_{th})}{\varepsilon_i^2} \frac{\partial^2 f_{th}}{\partial c_l \partial c_k}$$

have been dropped owing to the relatively small size of

$$\frac{1}{N} \sum \frac{(f_i - f_{th}(x_i))}{\varepsilon_i^2}$$

In general, the process of finding the coefficients involves expanding

$$\chi^2 = \chi_0^2 + \sum_k \frac{\partial \chi^2}{\partial c_{ok}} \delta_k + \frac{1}{2} \sum_{kl} \frac{\partial^2 \chi^2}{\partial c_{ol} \partial c_{ok}} \delta_k \delta_l$$

with $\delta_k = c_k - c_{ok}$, and solving for the δ's which predict zero partials of χ^2 by solving

$$\frac{\partial \chi^2}{\partial \delta_l} = 0 = \frac{\partial \chi^2}{\partial c_{ol}} + \frac{1}{2} \sum_k \frac{\partial^2 \chi^2}{\partial c_{ol} \partial c_{ok}} \delta_l \qquad \text{.....(7.8)}$$

$$\delta_l = - \sum_k \left(\frac{\partial^2 \chi^2}{\partial c_{ol} \partial c_{ok}} \right)_{lk}^{-1} \frac{\partial \chi^2}{\partial c_{ol}} \qquad \text{.....(7.9)}$$

so that the end of this process is a c_o for which $\frac{\partial \chi^2}{\partial c_{ol}} = 0$ and an inverse matrix of second derivatives at c_o. Having determined the coefficients that best describe the data, we now need to estimate the errors in these constants. Knowing these errors enables us to calculate uncertainties in other quantitites

such as peak positioning, width, and area.

Following the general prescription of Section 7.1, move f_p to $f_p+\Delta_p$ to yield a new χ^2, given by

$$\chi^2 = \chi_0^2 + \frac{\partial \chi_0^2}{\partial f_p} \Delta_p \qquad \text{.....(7.10)}$$

which can be expanded as

$$\chi^2 = \chi_0^2 + \sum_j \frac{\partial^2 \chi_0^2}{\partial f_p \partial c_j} \Delta_p \delta_j + \frac{1}{2} \sum_{ij} \frac{\partial^2 \chi_0^2}{\partial c_i \partial c_j} \delta_i \delta_j \qquad \text{.....(7.11)}$$

where the fact that $\frac{\partial \chi_0^2}{\partial c_j} = 0$ and $\frac{\partial^3 \chi_0^2}{\partial f_p \partial c_i \partial c_j} = 0$ has been used. The quantity χ^2 calculated using the point $f_p+\Delta_p$, minimizes at values of δ, that are different from those of zero already found for χ_0^2. These are found by setting $\frac{\partial \chi_0^2}{\partial \delta_l} = 0$ which yields

$$\delta_l = -\sum_k \left(\frac{\partial^2 \chi^2}{\partial c_{ol} \partial c_{ok}} \right)^{-1} \frac{\partial^2 \chi^2}{\partial f_p \partial c_k} \Delta_p \qquad \text{.....(7.12)}$$

from which we find

$$\frac{\partial c_l}{\partial f_p} = \frac{\delta_l}{\Delta_p} \qquad \text{.....(7.13)}$$

using equation 7.3 the error in c_l is

$$\left\{ (c_l - c_{lx(l)})^2 \right\} =$$

$$\left[-\sum_{k} \left(\frac{\partial^2 \chi^2}{\partial c_{ol} \partial c_{ok}} \right)^{-1} * - \sum_{m} \left(\frac{\partial^2 \chi^2}{\partial c_{ol} \partial c_{om}} \right)^{-1} \sum_{p} \left(\frac{\partial^2 \chi^2}{\partial f_p \partial c_m} \; \frac{\partial^2 \chi^2}{\partial f_p \partial c_k} \right) \varepsilon_p{}^2 \right] \quad \text{...(7.14)}$$

where the sum p over the data points has been interchanged with the matrix sums k and m. For this case of least-squares fitting, note that, from equation 7.5

$$\frac{\partial^2 \chi^2}{\partial f_p \partial c_j} = -2 \frac{\partial f_{th}(x_p)}{\partial c_j \varepsilon_p{}^2} \quad \text{.....(7.15)}$$

so that the sum over p in equation 7.14 is exactly twice $\frac{\partial^2 \chi^2}{\partial c_m \partial c_k}$ as given in equation 7.7, which allows the sum over m to yield δ_{lk} and a final result of

$$\left\{ (c_l - c_{lx(l)})^2 \right\} = \sigma_l{}^2 = 2 \left(\frac{\partial^2 \chi^2}{\partial c_{ol} \partial c_{ol}} \right)^{-1} \quad \text{.....(7.16)}$$

To recap, we have described the fitted function in terms of a set of constants and calculated the uncertainties in each of these constants. We must now use this information to calculate the error in the fit.

7.3 Error in the fitted function

Equations 7.3 and 7.4, along with equations 7.12 and 7.13, can now be used to find the errors in any function of the constants. One of the more interesting functions to consider is the fitted function, itself. The standard deviations in the constants are frequently much larger than the error in the function. That is due to the fact that large positive fluctuations in some constants will always occur with large negative fluctuations in others.

To begin

$$f(x) = f_{th}(c,x) \quad \text{.....(7.17)}$$

from which we calculate

$$\frac{\partial f(x)}{\partial f_p} = \sum_k \frac{\partial f_{th}(c,x)}{\partial c_k} \frac{\partial c_k}{\partial f_p} \qquad \text{.....(7.18)}$$

then using equation 7.3

$$\left\{(f(x)-f_{lx}(x))^2\right\} = \sum_{kl} \frac{\partial f_{th}(c,x)}{\partial c_k} \frac{\partial f_{th}(c,x)}{\partial c_l} \sum_p \frac{\partial c_k}{\partial f_p} \frac{\partial c_l}{\partial f_p} \qquad \text{.....(7.19)}$$

and using equations 7.12 and 7.13

$$= \sum_{kl} \frac{\partial f_{th}(c,x)}{\partial c_k} \frac{\partial f_{th}(c,x)}{\partial c_l} * \sum_m \left(\frac{\partial^2\chi^2}{\partial c_{ok}\partial c_{om}}\right)^{-1} * \sum_n \left(\frac{\partial^2\chi^2}{\partial c_{ok}\partial c_{on}}\right)^{-1}$$

$$* \sum_p \frac{\partial^2\chi^2}{\partial f_p \partial c_{om}} \frac{\partial^2\chi^2}{\partial f_p \partial c_{on}} \varepsilon_p^2 \qquad \text{.....(7.20)}$$

Then using equation 7.15 and equation 7.7 note that the sum over p is 2 times $\frac{\partial^2\chi^2}{\partial c_{om}\partial c_{ok}}$. Then the sum over n becomes δ_{lm} and the sum over m can be performed to yield

$$\left\{(f(x)-f_{lx}(x))^2\right\} = 2 \sum_{kl} \frac{\partial f_{th}(c,x)}{\partial c_k} \frac{\partial f_{th}(c,x)}{\partial c_l} \left(\frac{\partial^2\chi^2}{\partial c_k \partial c_l}\right)^{-1} \qquad \text{....(7.21)}$$

which may explain why the inverse matrix is sometimes called the error matrix. The off diagonal terms do the correcting for the fact that large and small coefficients are correlated. Note that many elements of the error matrix will normally be negative and will subtract from, rather than add to, the total error.

7.4 The error in the area of each peak

The area of each peak is given by height times width times a factor that is completely independent of statistics. Thus, only the error in $c^2w = c_1{}^2c_3$ needs to be determined here. Following the prescription of equation 7.3, we find

$$\frac{\partial c^2w}{\partial f_i} = 2cw\,\frac{\partial c}{\partial f_i} + c^2\,\frac{\partial w}{\partial f_i}$$

$$= 2c_1c_3\,\frac{\partial c_1}{\partial f_i} + c_1{}^2\,\frac{\partial c_3}{\partial f_i} \qquad \text{.....(7.22)}$$

We have assumed only a single peak, the square root of its height being the first constant, its location being the second constant, and its width being the third constant. In practice, there may be up to 27 more constants representing other peaks and up to 4 more constants representing the background contained in the matrix. From this we can extract the $\frac{\partial c}{\partial f_i}$ and $\frac{\partial w}{\partial f_i}$ needed to find the error in the peak area.

$$\left\{(c^2w\text{-}(c^2w)_{ex})^2\right\} = \sum_i \left(2c_1c_3\,\frac{\partial c_1}{\partial f_i} + c_1{}^2\,\frac{\partial c_3}{\partial f_i}\right)^2 \varepsilon_i{}^2$$

$$= (2cw)^2 \sum_{jk} \left(\frac{\partial^2\chi^2}{\partial c_1\partial c_j}\right)^{-1}\left(\frac{\partial^2\chi^2}{\partial c_1\partial c_k}\right)^{-1} 2\frac{\partial^2\chi^2}{\partial c_j\partial c_k}$$

$$+ 2(2cw)c^2 \sum_{jk} \left(\frac{\partial^2\chi^2}{\partial c_1\partial c_j}\right)^{-1}\left(\frac{\partial^2\chi^2}{\partial c_3\partial c_k}\right)^{-1} 2\frac{\partial^2\chi^2}{\partial c_j\partial c_k}$$

$$+ c^4 \sum_{jk} \left(\frac{\partial^2\chi^2}{\partial c_3\partial c_j}\right)^{-1}\left(\frac{\partial^2\chi^2}{\partial c_3\partial c_k}\right)^{-1} 2\frac{\partial^2\chi^2}{\partial c_j\partial c_k} \qquad \text{.....(7.23)}$$

so that

$$\left\{ (c^2w-(c^2w)_{ex})^2 \right\} = 2\left[(2cw)^2 \left(\frac{\partial^2\chi^2}{\partial c_1 \partial c_1} \right)^{-1} \right.$$

$$+ 2(2cw)c^2 \left(\frac{\partial^2\chi^2}{\partial c_1 \partial c_3} \right)^{-1}$$

$$\left. + (c^2)^2 \left(\frac{\partial^2\chi^2}{\partial c_3 \partial c_3} \right)^{-1} \right. \qquad \text{.....(7.24)}$$

Note that errors in the location of the peaks, in addition to the heights, locations, and widths of the other peaks, appear implicitly in the inverse matrix.

7.5 Inputting extra width information

The peaks of interest usually cannot be found to any great degree of accuracy by minimizing the χ^2 in equation 7.5. They are usually too close to other peaks or too close to background features, or simply too small. The same spectrum frequently contains other fitted peaks from which it is possible to find very accurate, but not perfect, estimates of the peak width. Since this peak width is, in general, due to detector response rather than the particular properties of the peak, it is quite reasonable to use this information in finding the position, height, and multiplicity of the peaks of interest. This is done by adding a penalty term to χ^2 so that it becomes

$$\chi^2 = \sum_i \left(\frac{f_i - f_{th}(c_n, x_i)}{\varepsilon_i} \right)^2 + \lambda(w_i - w_{ex})^2 \qquad \text{.....(7.25)}$$

where w_{ex} is the expected width. Because w_{ex} has an error, we do not want to force w_i to become exactly w_{ex}. We want the predicted error in w_i to be set so that the error in the peak area predicted by equation 7.24, and the error in the peak location predicted by equation 7.16, will be approximately correct.

Start by setting

$$A = \left(\frac{\partial^2 \chi^2(\lambda=0)}{\partial c_1 \partial c_1} \right)$$

$$B = \left(\frac{\partial^2 \chi^2(\lambda=0)}{\partial c_1 \partial c_3} \right)$$

$$D = \left(\frac{\partial^2 \chi^2(\lambda=0)}{\partial c_3 \partial c_3} \right) \qquad \text{.....(7.26)}$$

and note from equation 7.25, that for λ not equal to zero

$$D + 2\lambda = \left(\frac{\partial^2 \chi^2(\lambda=0)}{\partial c_3 \partial c_3} \right) \qquad \text{.....(7.27)}$$

For an isolated peak whose error comes from its height and width alone, it is useful to note that

$$\left(\frac{\partial^2 \chi^2}{\partial c_i \partial c_j} \right) = \begin{pmatrix} A & B \\ B & D+2\lambda \end{pmatrix} \qquad \text{.....(7.28)}$$

and

$$\left(\frac{\partial^2 \chi^2}{\partial c_i \partial c_j} \right)^{-1} = \frac{1}{A(D+2\lambda)-B^2} \begin{pmatrix} D+2\lambda & -B \\ -B & A \end{pmatrix} \qquad \text{.....(7.29)}$$

This enables us to calculate λ by noting that

$$\delta^2 w = \frac{2A}{A(D+2\lambda)-B^2} \qquad \text{.....(7.30)}$$

and thus the error matrix is

$$\left(\frac{\partial^2\chi^2}{\partial c_i \partial c_j}\right)^{-1} = \begin{pmatrix} \frac{1}{A}+\delta^2 w\frac{B^2}{2A^2} & \frac{-B\delta^2 w}{2A} \\ \frac{-B\delta^2 w}{2A} & \frac{\delta^2 w}{2} \end{pmatrix} \qquad \text{.....(7.31)}$$

This approximation which is used by ROBFIT is not exact. The penalty term in χ^2 is actually independent of f_i so that

$$\left(\frac{\partial^2\chi^2}{\partial f_i \partial c_j}\right) = \left(\frac{\partial^2\chi^2(\lambda=0)}{\partial f_i \partial c_j}\right) \qquad \text{.....(7.32)}$$

Thus in equation 7.14, the inverse matrices are $\left(\frac{\partial^2\chi^2(\lambda)}{\partial c_l \partial c_m}\right)$ while the sum over the data points produces $\left(\frac{\partial^2\chi^2(0)}{\partial c_m \partial c_k}\right)$ which prevents the final matrix reductions so that our overall error matrix becomes

$$E_{ij} = \left(\frac{\partial^2\chi^2(\lambda)}{\partial c_i \partial c_l}\right)^{-1}\left(\frac{\partial^2\chi^2(\lambda)}{\partial c_k \partial c_l}\right)^{-1}\left(\frac{\partial^2\chi^2(0)}{\partial c_l \partial c_j}\right) \qquad \text{.....(7.33)}$$

or

$$E = \frac{1}{(A(D+2\lambda)-B^2)^2}\begin{pmatrix} A(D+2\lambda)^2-2B^2(D+2\lambda)+B^2D & B^3-BAD \\ B^3-BAD & A(AD-B^2) \end{pmatrix} \text{.....(7.34)}$$

rather than equation 7.29. The λ in equation 7.32 is quite different from the λ in equation 7.29. Again, solving for λ by setting $\delta^2 w$ to $2E_{33}$, the error matrix in terms of $\delta^2 w$ becomes

$$E = \begin{pmatrix} \frac{1}{A}+\delta^2 w\frac{R^2}{2A^2} & \frac{-B\delta^2 w}{2A} \\ \frac{-B\delta^2 w}{2A} & \frac{\delta^2 w}{2} \end{pmatrix} \qquad \text{.....(7.35)}$$

which is exactly the same as before. The only difference is the value of λ

$$\lambda_a = \frac{B^2-AD}{2A} + \frac{1}{\delta^2 w} \qquad \text{.....(7.36)}$$

while for the true error matrix

$$\lambda_t = \frac{B^2-AD}{2A} + \sqrt{\frac{2(AD-B^2)}{A\delta^2 w}}$$

which, because of the square root, is much smaller than the approximate value. Putting the approximate λ_a into equation 7.34 for $\delta^2 w$ yields

$$\delta^2 w = \frac{(AD-B^2)}{A}\,\delta^4 w_a = \frac{\delta^4 w_a}{\delta_N^2}$$

which is the actual spread of width values that will be found when ROBFIT uses equation 7.30 to adjust the penalty function.

In summary, width information is put into the error matrix by adding a penalty function. The resulting width error, as calculated from the inverse matrix of a two by two matrix involving the peak's height and width, then correctly predicts the two by two error matrix. It is assumed that this same property survives the much larger matrix inversions made in practice. The principle caveat is that the output widths from this procedure will not deviate from the input widths by more than the assumed error, even though the error matrix will be correct.

8 Application of ROBFIT

This chapter has been included to give the reader a feel for how ROBFIT is used to analyze a spectrum. The first section shows the accuracy that can be achieved when dealing with clearly separated peaks. It also outlines some of the practical considerations that must be addressed before fitting is started. Section 8.2 gives a detailed account of the re-analysis of the supernova data. It shows how ROBFIT can be tuned to a particular analysis method. The re-analysis illustrates how easy it is to fit a spectrum, even if peak shapes and widths are unknown prior to fitting. The final section gives a feel for how the code can be used on different kinds of spectra. The example we show is a study of solar seismology data. These spectra do not follow the usual Gaussian statistic fluctuations from channel to channel. We show that the code can effectively deal with this kind of data even though ROBFIT's minimization scheme has been designed around Gaussian statistics.

8.1 Computer-generated test cases

The following computer-generated test cases have been performed to provide an insight into the operational accuracy of ROBFIT. They test the codes ability to correctly reproduce a single peak in both high and low background regions. The tests also study the effect of peak width on determination of peak parameters. The code's accuracy in detecting multiple peaks is also addressed. For the purpose of these tests, it is worth clarifying the definition of background. The background is defined as the underlying continuum on which the peaks sit. Any additional peaks, introduced by fluctuations in this background continuum, will be called spurious peaks even though they are, indeed, a background to the true peaks.

8.1.1 Peaks in large background regions

A constant background level of 50 counts is used in the following test runs. Peaks are then added to this background by specifying their position, width and total area under the peak. The number of counts in each channel has been randomly distributed using a Gaussian function. The standard deviation in a channel has been set equal to the square root of the number of counts in the channel.

8.1.1.1 Large peaks

A Gaussian shaped peak has been generated in this test. It has a height of approximately four times the background level. Each peak has been positioned

TABLE 8.1 Results of five separate fits to large peaks.

Fit	Position (channels)	Error (channels)	Width (channels)	Error (channels)	Strength (counts)	Error (counts)
1	500.55	0.39	58.38	0.82	9649	135
2	500.24	0.41	59.69	0.86	10118	145
3	500.28	0.38	60.91	0.80	10216	134
4	499.80	0.37	59.33	0.78	9871	129
5	499.56	0.40	58.44	0.85	9932	145

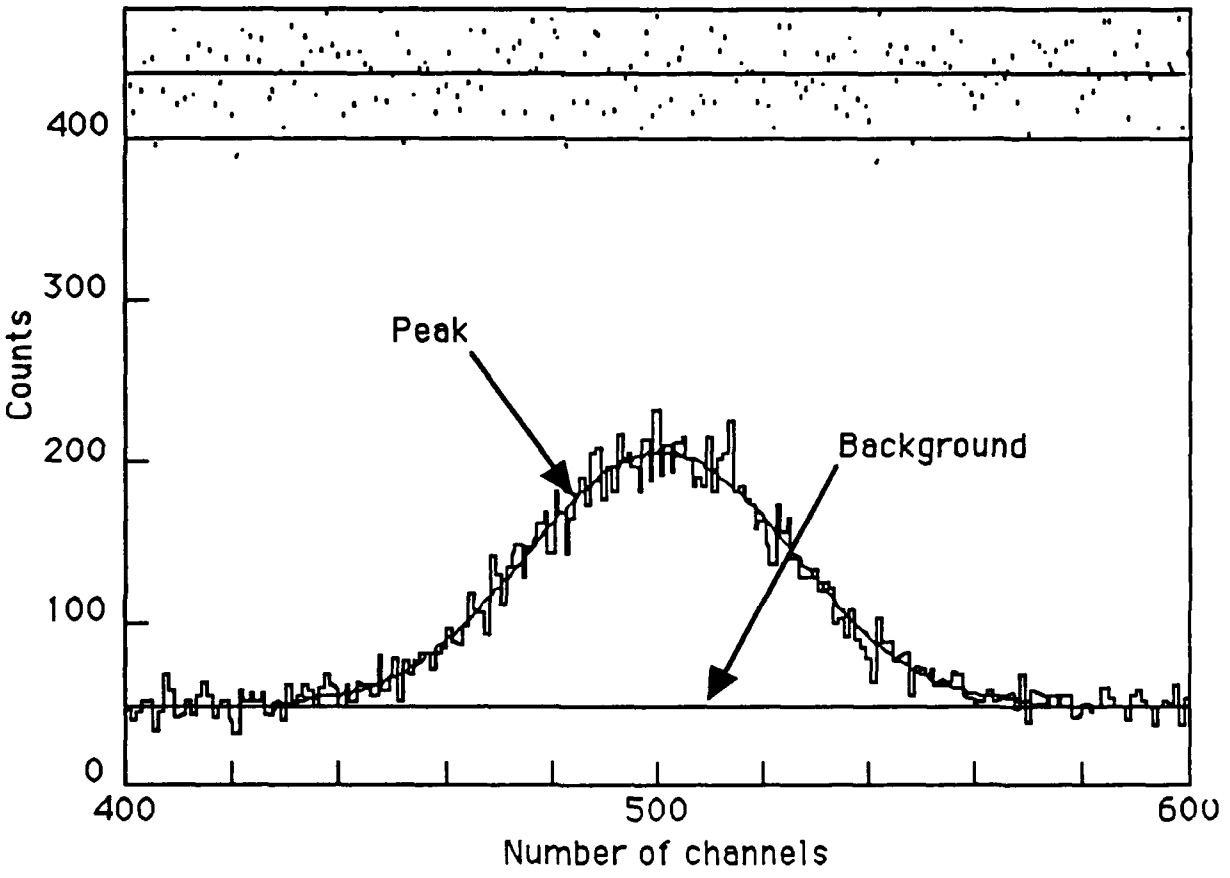


Figure 8.1 Result of a fit to a large peak on a large background. For peak parameters see fit 1 in Table 8.1.

at channel 500 in a 1000 channel spectrum, with a width of 58.85 channels. The area under each peak, or strength, is 10000 counts. Table 8.1 shows the results of fits to five separate data sets where the position/error in position, width/error in width, and strength/error in strength are given for each peak.

This simple test shows that the program can accurately reproduce the underlying peak function. The results of fit 1 are shown in Figure 8.1.

8.1.1.2 Small peaks

To test the code's ability to detect small peaks on a large background we show the results of five separate fits to the Gaussian shaped peak. In these tests the total area underneath the peak has been reduced to 700 counts. This gives a peak height of 11 counts or approximately 1.6 times the background fluctuation level. Fitting peaks of this size requires a low CUTOFF; we have used a value of 2. At this level one anticipates finding a few spurious peaks. The number of acceptable spurious peaks is entirely dependent on the requirements of the data analysis. Here we have a maximum of one spurious peak in every 100 channels. The results of the five test runs are detailed in Table 8.2, and fit 1 is shown in Figure 8.2.

Table 8.2 Fit to peaks the same size as the background fluctuations.

Fit	Position (channels)	Error (channels)	Width (channels)	Error (channels)	Strength (counts)	Error (counts)
1	504.54	3.31	51.83	5.12	612	74
2	494.59	3.95	45.00	6.96	667	122
3	503.16	3.23	62.64	5.16	853	83
4	497.46	3.40	55.63	5.00	653	75
5	497.75	3.30	52.65	5.62	704	84

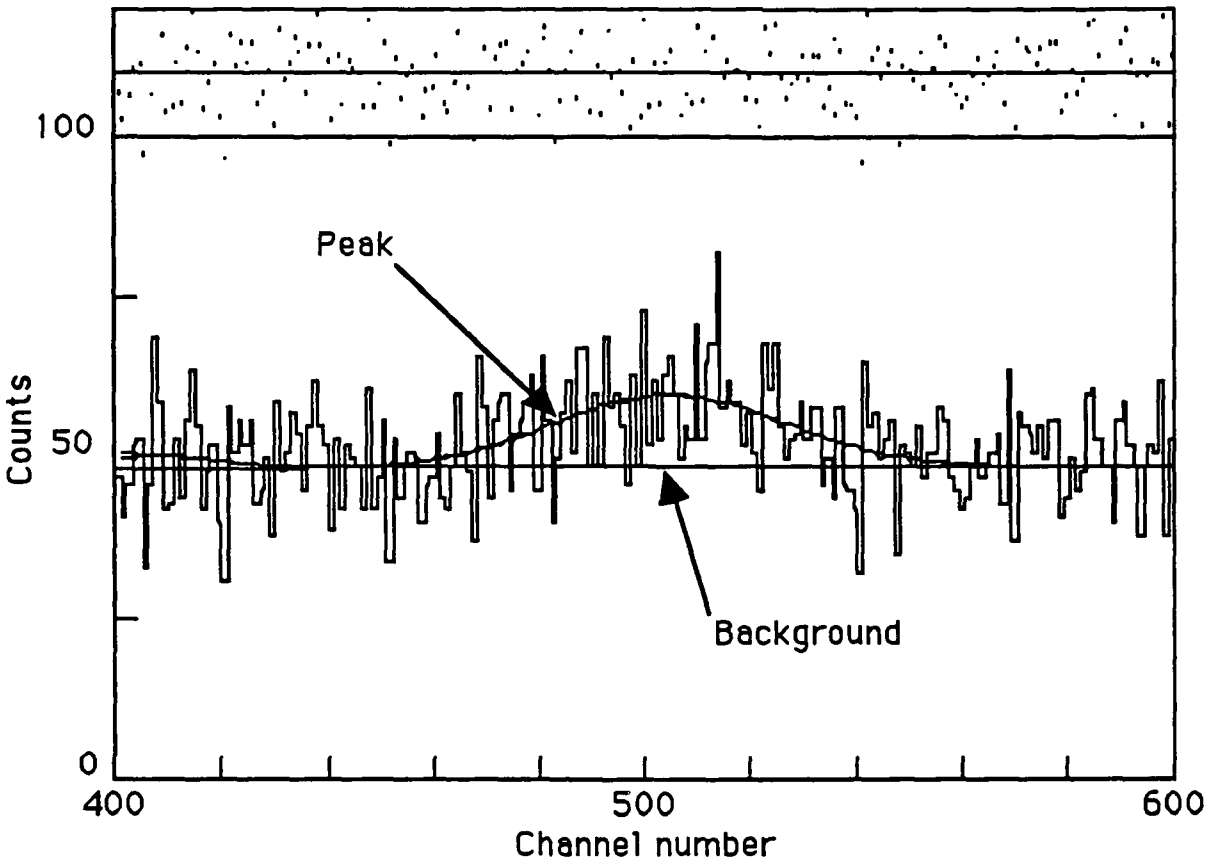


Figure 8.2 Result of a fit to small peak on a large background. For peak parameters see fit 1 in Table 8.2.

8.1.2 Peaks in low background regions

The following sections examine the detection efficiency of ROBFIT when the background under the peak is small. In these tests a background level of 2 has been used. Again, peaks are added and the counts in each channel distributed according to a Gaussian distribution as described in Section 8.1.1.

8.1.2.1 Large peaks

Here we chose a Lorentzian to be the peak shape and generated three peaks at channel positions 167.6, 501 and 834.3. The peaks have half widths of 11.11, 6.67 and 4.45 channels, and the areas under each peak are 6824, 4000 and 2000 respectively. These parameters give each peak a height of approximately 200 counts. Table 8.3 shows the results of two fits to separately generated data sets, and Figure 8.3 shows the fit to the first set of three peaks.

Table 8.3 Fit to large peaks on a low background.

Fit	Peak	Position (channels)	Error (channels)	Width (channels)	Error (channels)	Strength (counts)	Error (counts)
	1	167.44	0.20	22.49	0.41	6830	93
1	2	501.18	0.16	13.10	0.32	3791	70
	3	834.62	0.16	8.81	0.32	1902	51
	1	167.78	0.21	22.15	0.41	6777	95
2	2	500.88	0.15	12.52	0.30	3982	73
	3	834.35	0.15	8.42	0.29	1953	51

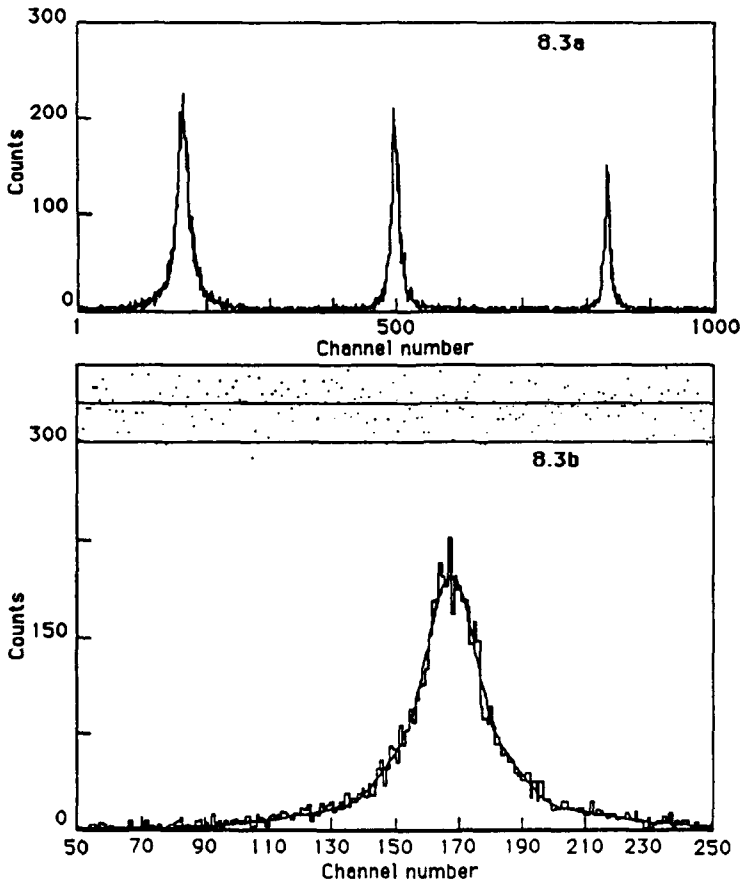


Figure 8.3 Results of fit 1 of Table 8.3 (a) three large Lorentzian shaped peaks on a low background and (b) the fit to the lowest channel region.

8.1.2.2 Small peaks

To test ROBFIT's ability to detect small peaks on a low background, we have generated three Lorentzian shaped peaks in a manner similar to that described in Section 8.1.2.1. In these tests, each peak has the same area; 50 counts. This gives the peaks heights of 1.4, 2.4, and 3.6 counts respectively. These peak heights are to be compared with a background level of 2 ± 1 counts. The results of these low peak, low background tests are shown below in Table 8.4 and in Figure 8.4.

Table 8.4 Fits to small peaks on a low background

Fit	Peak	Position (channels)	Error (channels)	Width (channels)	Error (channels)	Strength (counts)	Error (counts)
1	1	165.79	4.23	26.40	5.50	70	18
	2	507.15	3.66	19.31	5.50	53	16
	3	836.70	1.42	7.88	3.50	42	13
2	1	171.50	5.00	23.20	5.59	50	17
	2	500.81	1.47	11.17	3.59	71	16
	3	834.53	0.94	6.58	2.27	54	13

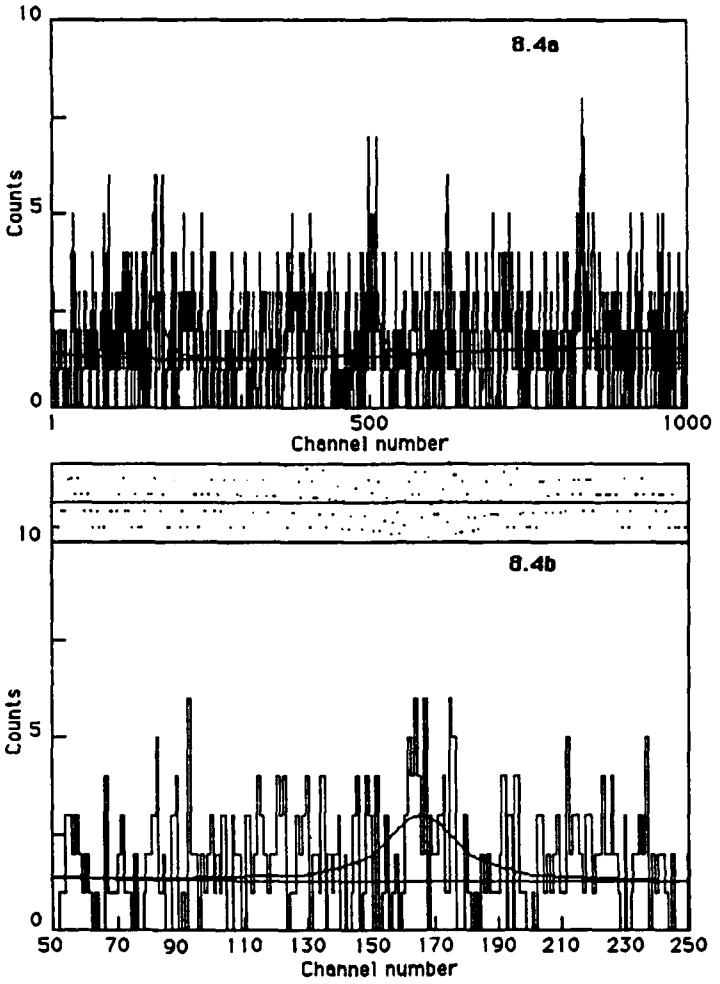


Figure 8.4 Results of fit 1 of Table 8.4. (a) Small Lorentzian shaped peaks on a low background, and (b) the fit to the lowest channel region.

8.1.3 Detecting peaks with narrow widths

When dealing with histogrammed data, the user must always be aware of the effects of binning on the peak detection mechanism. In a typical nuclear physics experiment, data taking is usually organized so that spectral features have widths greater than 3 or 4 channels. Even at this binning the histogram does not contain all the information of the underlying deriving function[23]. However, it may be that a finer binning is not possible. This means the analysis technique must be capable of dealing with these narrow peaks.

This section shows how accurately ROBFIT can fit narrow peaks. A Gaussian shaped peak with a strength of 10000 counts and a width of 5.86 channels has

been used as the test peak. The peak is placed upon a constant background level of 50 counts.

In the first test, the centroid of the peak is at channel 500. Table 8.5 shows the results of three separate fits to this data. The results of the first fit are shown in Figure 8.5.

Table 8.5 Fits to a peak centered on channel 500

Fit	Position (channels)	Error (channels)	Width (channels)	Error (channels)	Strength (counts)	Error (counts)
1	500.01	0.03	5.98	0.06	9874	112
2	500.01	0.03	5.94	0.05	10209	110
3	500.02	0.03	5.94	0.06	10006	125

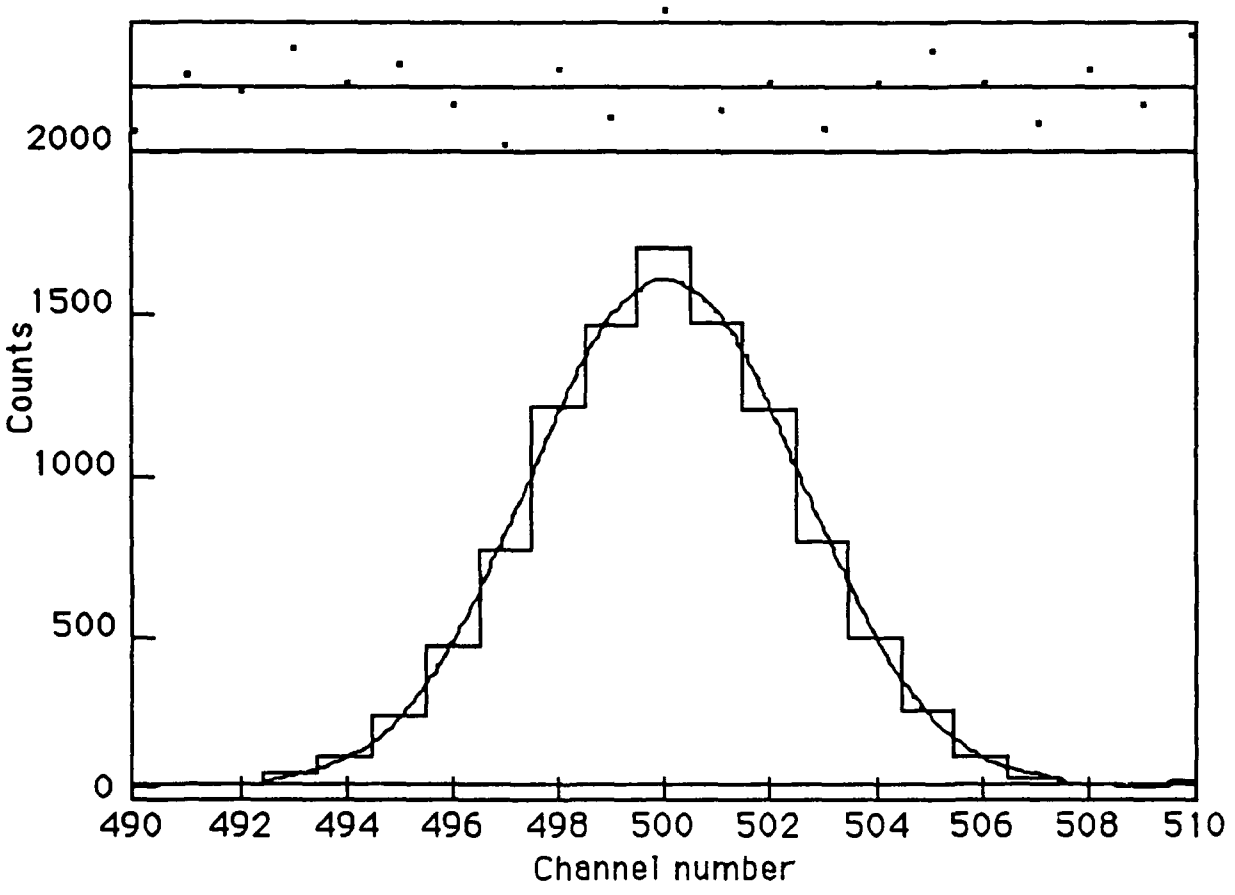


Figure 8.5 Fit to a peak centered at channel 500, using the data from fit 1 of Table 8.5.

To examine ROBFIT's sensitivity to shape changes due to the binning boundaries, the second test has the centroid of the peak shifted to position 500.3. Table 8.6 gives the results of three fits to this data. The results of fit 1 are shown in Figure 8.6.

Table 8.6 Fits to a peak centered on channel 500.3

Fit	Position (channels)	Error (channels)	Width (channels)	Error (channels)	Strength (counts)	Error (counts)
1	500.32	0.03	5.98	0.06	9878	115
2	500.30	0.03	5.94	0.05	10209	115
3	500.32	0.03	5.95	0.06	10016	119

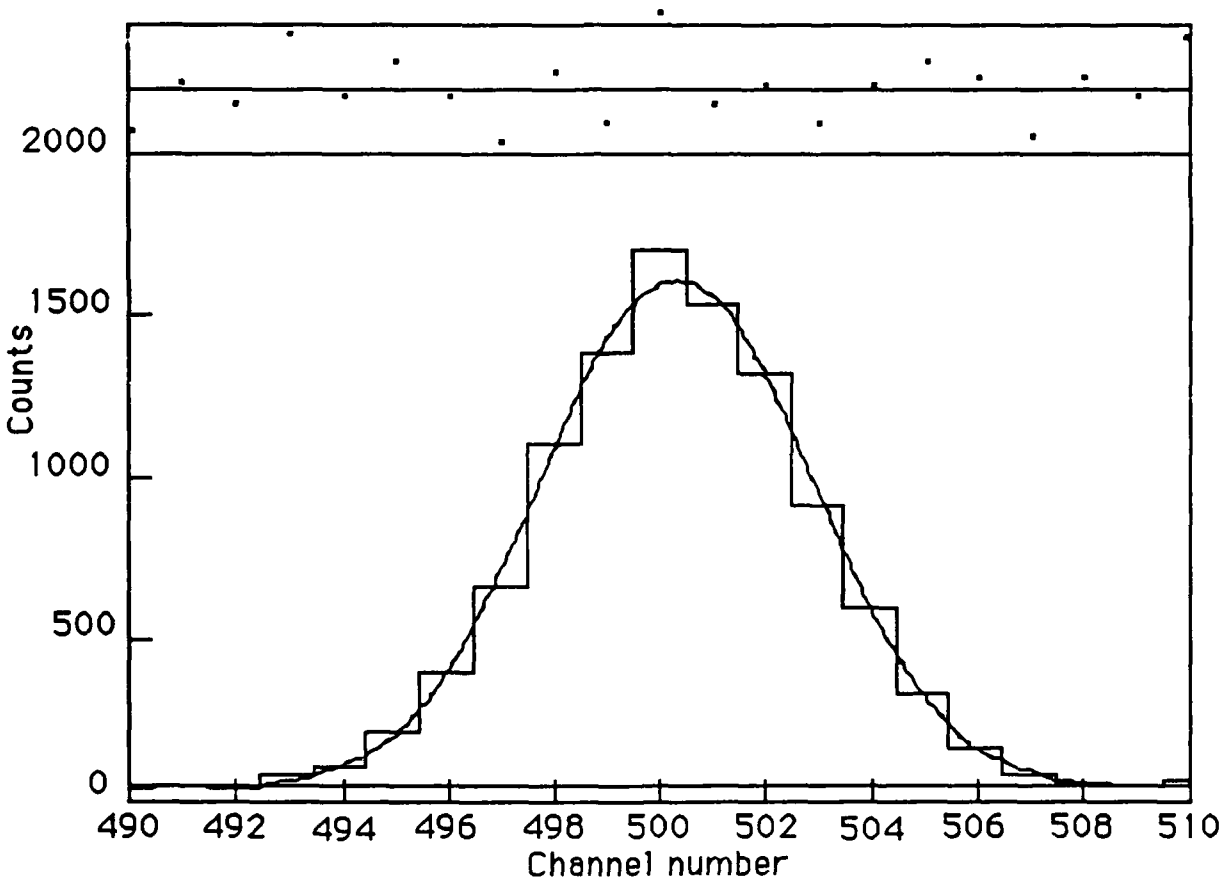


Figure 8.6 Fit to a peak centered at channel 500.3 using data taken from fit 1 of Table 8.6.

The peak detection performance of the code at these narrow peak widths is seen to be the same as that at the wide widths of section 8.1.1 and 8.1.2.

8.2 Supernova data

In this section we give a detailed account of the analysis of the supernova data. The results of this analysis have been presented in reference 1. Here we give the step-by-step reduction of the raw data. This should provide the reader with an insight into how ROBFIT can be used in the analysis of real data. Figure 8.7 shows the raw data. We have concentrated our analysis on the channel range 57 to 4096; the first 57 channels being used for timing signals and threshold offsetting.

In this analysis we will need to set a number of startup parameters. These must be defined at the outset of each run. The first step of the data reduction will illustrate how these parameters are presented in this section.

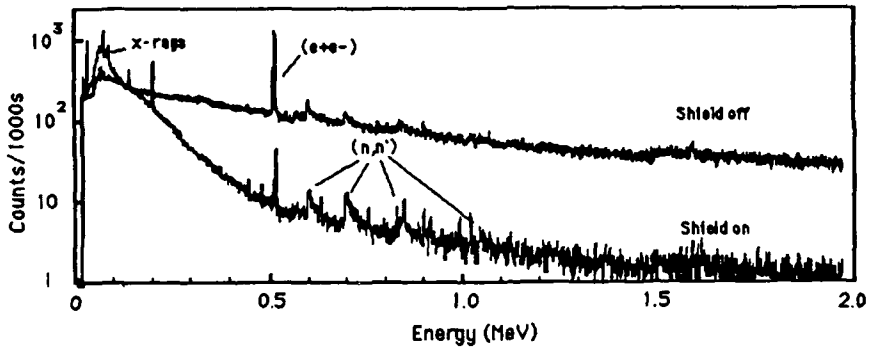
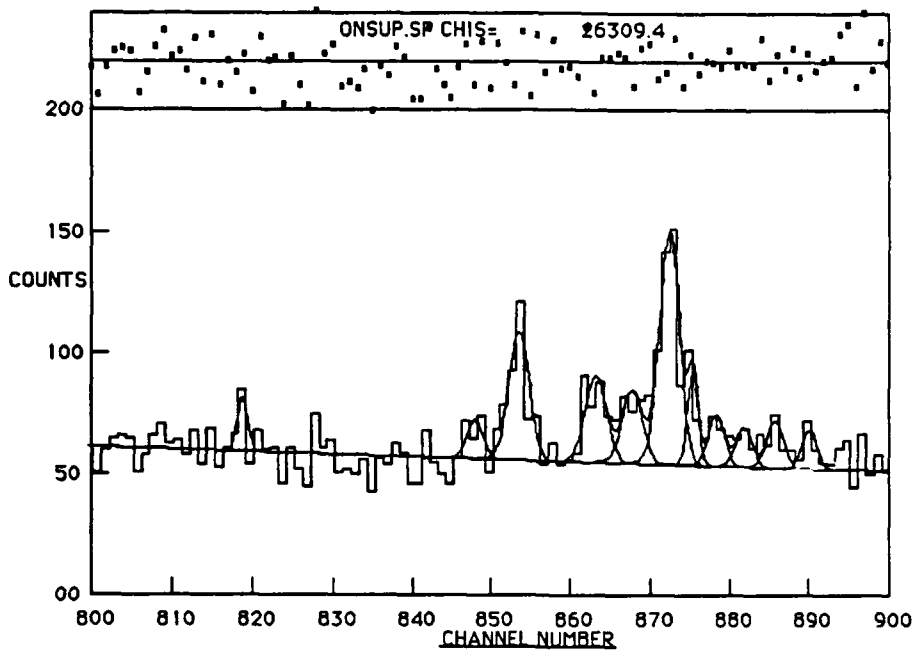


Figure 8.7 The supernova gamma-ray spectrum with the veto shield on, and off. The analysis in this chapter has been performed on the "shield on" spectrum. Indicated in this figure are the four neutron inelastic scattering peaks (n,n').

In the first step we are only concerned with getting a rough description of the peak shapes within the spectrum. To begin, we have generated a Gaussian standard containing five back-to-back cubic splines. This standard is then introduced to ROBFIT as the description of the peak shapes within the spectrum. From a brief inspection of the raw data we have estimated the peak widths and for this first run we have set the width range to be from 5 ± 2 at channel 520 to 5 ± 2 at channel 1050. We expect to be able to locate all peaks to within 3σ of the background fluctuations by specifying a CUTOFF range from



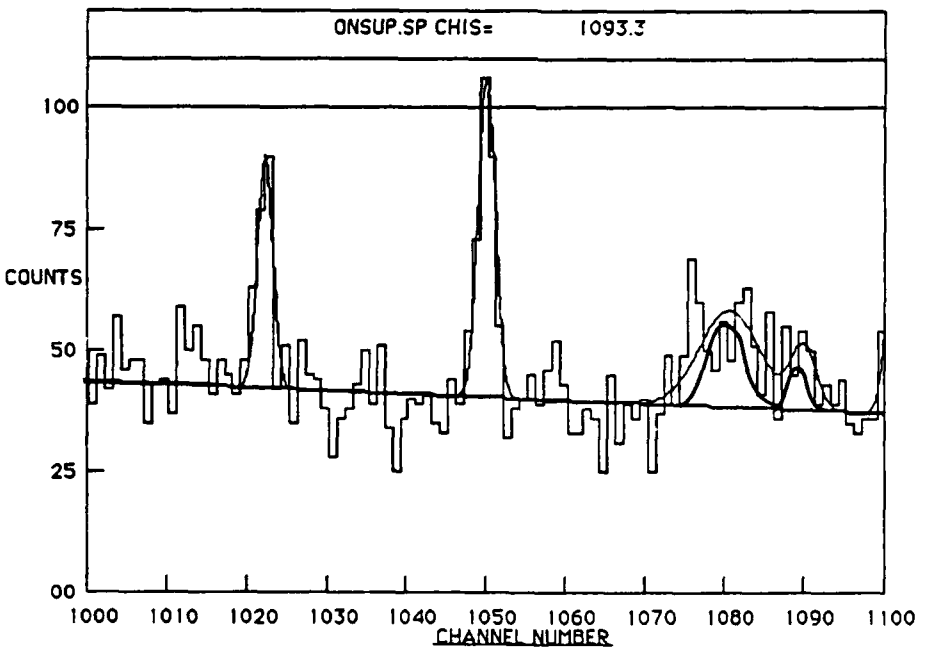


Figure 8.8 a) above and b) below Fits to the 700-800 and 1000-1100 channel regions of the supernova data. The figures illustrate the two peak shapes that are present in the supernova data.

10 to 3, taking 7 steps between these limits. At this point these values are just convenient starting levels. We do not expect to be able to identify all the peaks until we have representations of all the peak shapes in the data. There is one last parameter that must be specified, and it is the number of coefficients to be fitted to the background. Again, as a starting point, we will begin the background cycle with 4 coefficients and add to it up to a maximum of 8. The above starting conditions are defined by:

4,8	Background constant range
10,3,7	CUTOFF range
5,2,520,5,2,1050	The single standard width range
$^{W}L,^{E}W,^{P}L,^{W}M,^{E}M,\ ^{P}M$	Additional width variations for other standards specified as low and high widths, errors and positions.

Figure 8.8 illustrates the types of peaks identified in this first trial run and Table 8.7 lists all the fitted peak parameters. As can be seen, the single peaks at channels 1022 and 1050 have been well fitted. The broad regions between channels 860 and 890, and 1080 and 1090, where we know neutron inelastic scattering peaks reside, have been fitted with multiple peaks. This multiple peak fitting is called braiding and it generally signifies that we have an incorrect peak shape. Of course, if there really was only one peak shape present within the spectrum, this braiding would be justified and the user would need to study these regions carefully to understand the overlapping peaks. In this data we know *a priori* that these regions actually require a fundamentally different peak shape. These regions are dominated by neutron inelastic scattering peaks. The spectrum contains four such regions centered around channels 620, 720, 870 and 1080. We must, therefore, generate a peak standard to take care of these shapes. This becomes important later in the analysis because one of the supernova peaks actually resides on top of the 870 neutron peak. How do we create the standard for these peaks? Because we have no analytic function for generating these shapes, we must create one from the data. For this purpose the neutron peak within the channel range 700 to 760 was fitted with 5 back-to-back cubic splines. This region of the spectrum is shown in Figure 8.9. This second standard can now be entered into the fitting. Before starting the second round of fitting we must also think

about our Gaussian shaped standard. Is this shape ideal for the gamma-ray peaks? A well defined peak shape produces a better fit. Before the supernova experiment was undertaken, a number of gamma-ray lines were measured with the detector that was to be used in the experiment. By doing this, a large number of counts could be accumulated under one gamma-ray peak. Then, using the standard fitting routine, a precise standard could be made. This new standard is called the YT standard after the Yttrium spectrum from which it was generated. The YT standard can now replace the Gaussian standard used in the first round of fitting.

Table 8.7 ROBFIT output from the first round of fitting

```
OUTPUT FROM ROBFIT
IP=    1
  27 PEAKS 1001 CH#N, CHIS=       1093. NITB=    8
CUTOFF=   3.00
 1.0672  1.0000
 526.356    .081      3.704    .170       1854.80     84.12  1
 586.144    .847      8.344   1.958        255.97     52.91  1
 618.965    .491      9.000   1.133        896.02     99.80  1
 631.662   1.194      9.000   3.873        339.37    122.93  1
 642.382   2.205      7.017   3.876        135.41     84.02  1
 648.958    .334      3.255    .778        221.32     54.28  1
 718.864    .413      7.862    .943        836.83     86.38  1
 726.379    .784      4.283   2.230        202.49    216.60  1
 732.597   1.821      9.000   5.748        440.47    281.39  1
 744.543   1.352      9.000   2.713        347.65    108.62  1
 777.033    .231      3.143    .536        168.98     25.42  1
 853.649    .245      2.926    .541        160.91     27.26  1
 864.857    .780      6.735   1.983        198.82     47.60  1
 872.591    .244      4.085    .574        352.49     47.99  1
 882.420   1.312      9.000   2.592        153.97     42.93  1
 912.615    .576      4.149   1.326         86.35     24.40  1
 927.202    .199      2.520    .445        171.22     27.87  1
 934.195   1.487      5.641   2.665         63.90     31.84  1
 942.405    .343      2.749    .774        102.38     25.83  1
1022.343    .200      2.197    .444        112.20     20.71  1
1050.151    .137      2.452    .299        173.82     19.94  1
1080.635    .932      9.000   2.363        191.62     41.12  1
1089.929    .911      3.767   2.120         51.03     28.29  1
1101.082    .289      2.311    .661         62.17     15.94  1
1157.174    .255      1.515    .489         39.22     12.49  1
1263.145    .399      2.205    .919         38.37     14.18  1
1418.439    .763      4.445   1.766         48.29     16.91  1
```

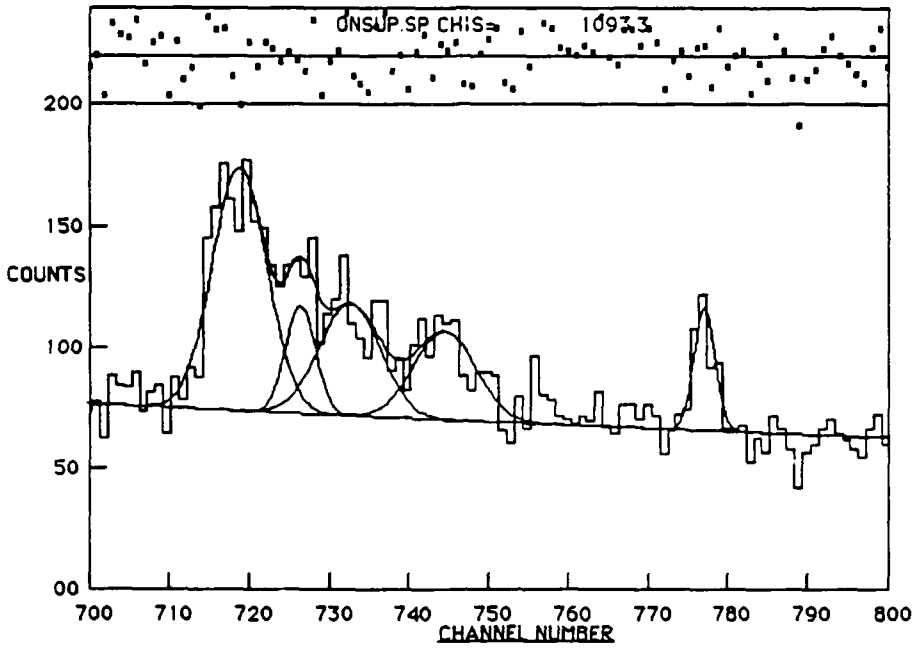


Figure 8.9 The region of the spectrum used to generate the neutron inelastic scattering peak. The fit is from the first set of fitting.

The new standards are introduced in step 2 of the fitting. The starting parameters for this step are

4,12
10,3,7
2.5,0.5,927,2.5,0.5,1050 YT Standard
11,2,620,25,2,867 Neutron Standard

We have allowed four more constants to be added to the background and the CUTOFF range is held the same as the first two runs. We have now included both standards in the fit. All the braiding peaks must be removed from the peak list before we begin the fit.

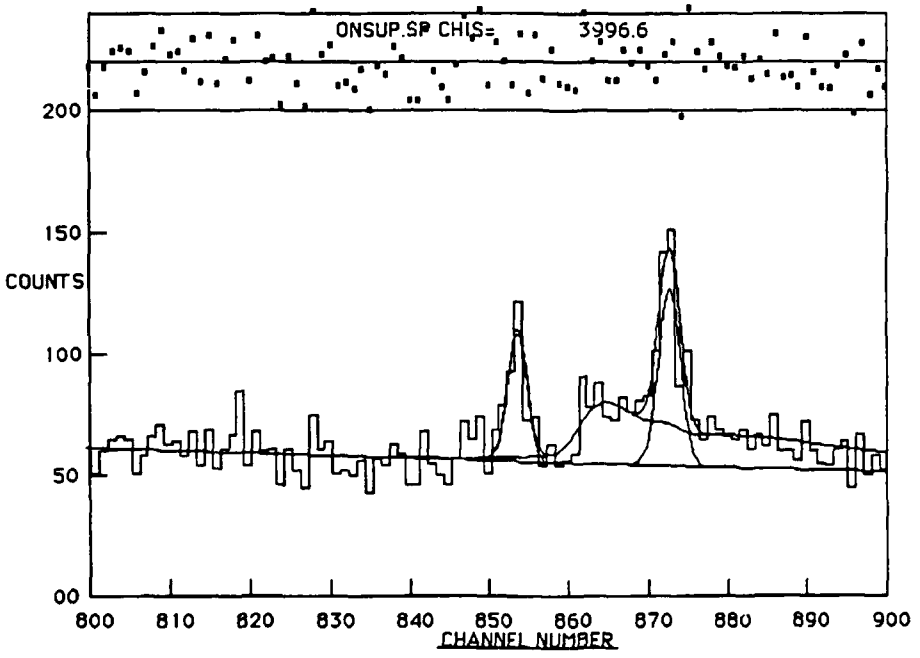


Figure 8.10 Third round fit to the third inelastic scattering peak.

Figure 8.10 shows the fit to the neutron peak at channel 870. The braiding has now been corrected, but on closer inspection the width of the neutron inelastic scattering peak is seen to be too large. Upon inspection of the other inelastic scattering peaks we find that the peak in channel region 720 seems to be fitted resonably well. Therfore, we take its width as the starting point for the next step.

Determining the correct neutron peak widths is the objective of step 3. The starting parameters are

4,12	
10,10,1	
2.61,0.36,927.2,2.47,0.28,1050.2	YT standard
11,1,615.8,11,1,864	Neutron standard

where we have re-defined the neuton standard widths.

Because the previous run had a CUTOFF level of 3σ, and has thus found most of the peaks present in the spectrum, all we needed to do in this run was re-fit the peaks that are already present. Consequently, we ran the CUTOFF to a level of 10σ. The result of these re-fits are shown in Figure 8.11. It can be seen that the neutron peaks have been determined quite well. Most important for the supernova analysis is the fact that the 870 neutron peak has been well fitted. We are still having some problems with the 1080 channel neutron peak; however, for the remainder of this analysis the precise fitting of this region is irrelevant.

When looking for a good fit it is best to study the residuals plotted above each figure. For a good fit, the points must be equally scattered above and below the center line. Low points generally means the background level is too high, and usually indicates undetected peaks within that region. As these peaks are identified, the background level will drop.

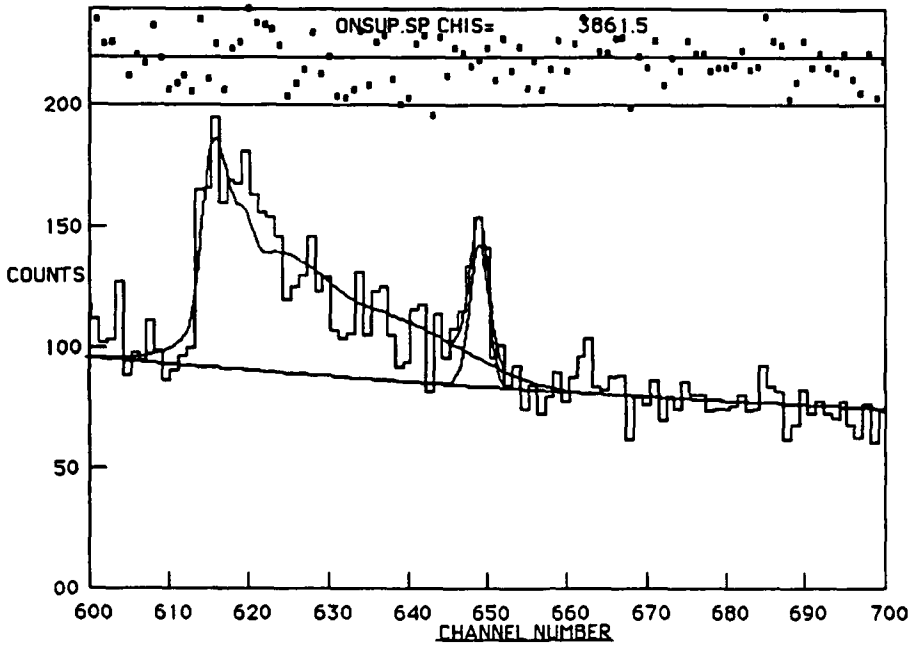


Figure 8.11a The 620 channel region after the third round of fitting.

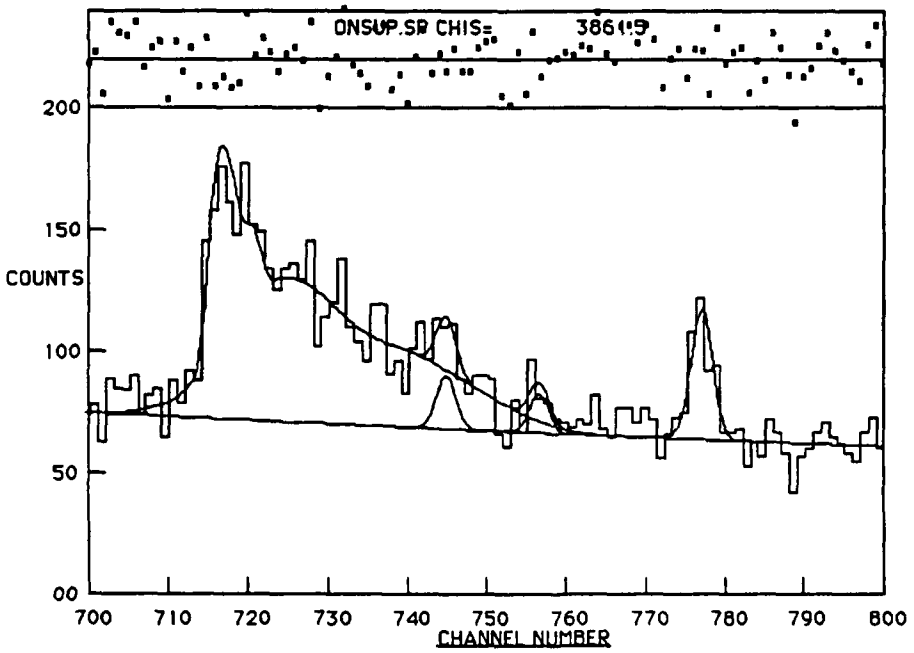


Figure 8.11b The 720 channel neutron region after the third round of fitting.

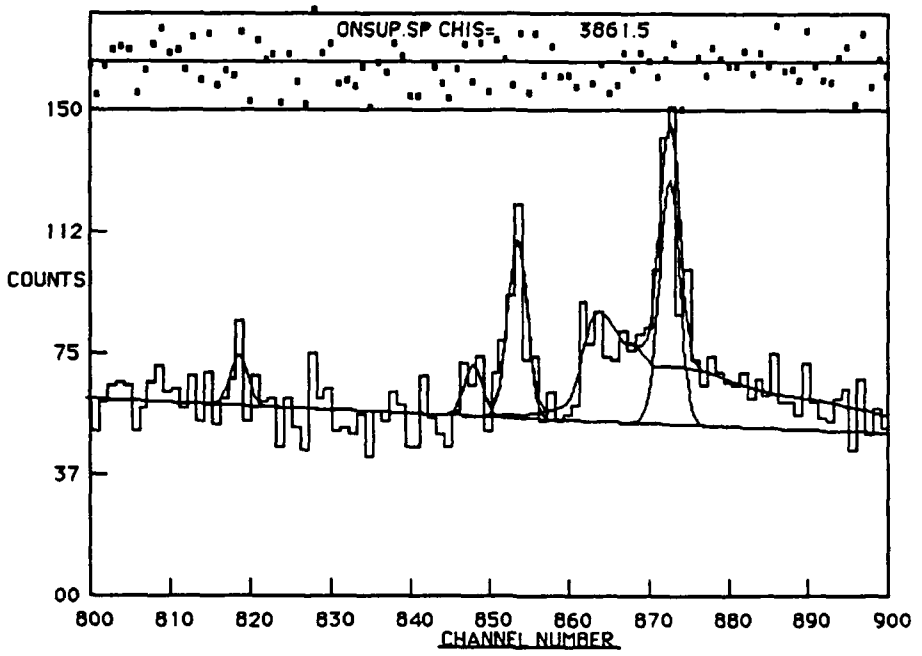


Figure 8.11c The 870 channel neutron region after the third round of fitting.

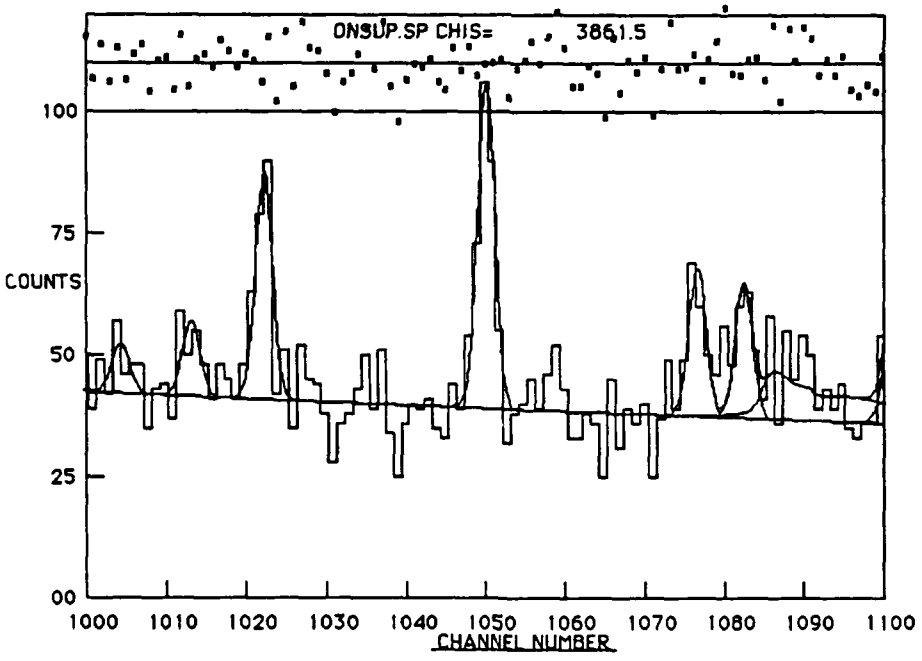


Figure 8.11d The 1080 channel neutron region after the third round of fitting.

Having fitted the neutron inelastic scattering peaks and the gamma-ray peaks to reasonable accuracy, we can start the search for the supernova signals. Let us concentrate on the high channel region first. We expect to see a signal around the 1260-1290 channel region. The first thing we find is that we must reduce the CUTOFF level below 3σ in order to pick up any features in this region. Although running the CUTOFF below 3σ will allow spurious peaks to enter into the peak list, any fitted peaks will be supplied with an associated error. From this the user can decide if the peak is actually a real signal. Of course, the user must monitor the overall χ^2 value for 'goodness' of fit, remembering that the χ^2 per number of degrees of freedom should be around 1. We do not want to introduce too many peaks or spurious over-fitting will occur.

Step 4 is performed to reduce the CUTOFF level; the starting values being

4,12
10,2,8 for a first run then 2,1.9,1 for the second
2.61,0.36,927.2,2.47,0.28,1050.2
12.8,0.6,717,12.8,1.0,863

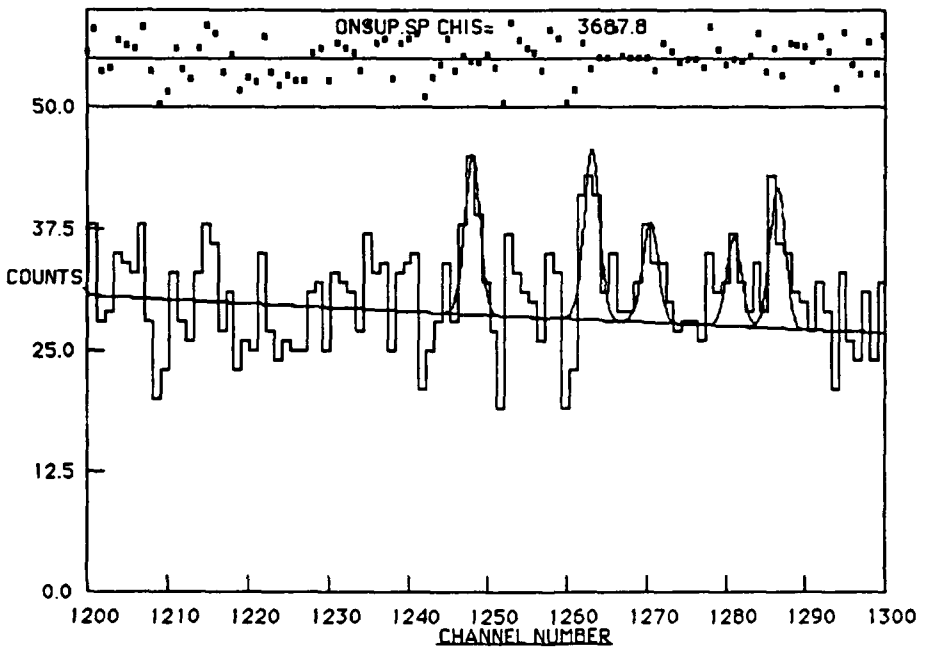


Figure 8.12 High supernova region after the fourth round of fitting.

We have used the neutron widths from step three to refine the width range. We have performed two succesive runs of this width setup; the first was to a 2σ CUTOFF, and the second ran the cutoff to an even lower level of 1.9σ. This was because even at the 2σ level the high supernova clump was not found. The 1.9σ fit is shown in Figure 8.12. The low supernova region at channel 870 is identical to the fit in Figure 8.11.

At this point we could end the analysis and conclude that there is no supernova peak at the low region and that four gamma-ray lines have been identified at the high region. The rest of the analysis would then involve attempting to give meaning to these lines. However, from a second set of data taken when the experiment was not fully operational, but good enough to give some estimate of the shape of these supernova peaks, there was an indication that the supernova lines may be somewhat broader than the standard gamma-ray line. This is not too difficult to imagine considering that the supernova is an explosion with fragments speeding in all directions. This motion would impart a spread of velocities on any gamma-ray line originating from this debri which would broaden the line. The following analysis not only gives a deeper insight into the dynamics of the supernova but also illustrates the flexibility of ROBFIT. Assuming that any peak due to the supernova is going to be broad in nature, we can generate a third standard, which we will call the wide line standard, to take care of this shape. The standard is a Gaussian shape and is composed of three back-to-back cubic splines.

In step 5 we remove all the peaks that have been fitted to the high supernova region and introduce the wide line standard. The starting parameters are:

```
4,12
3,2,2
2.61,0.36,927.2,2.47,0.28,1050.2
12.8,0.6,717,12.8,1.0,863
6,2,1260,6,2,1285                    wide line standard
```

where the width limits of the third standard have been estimated from the data structure around the 1280 region. At this point they are just a convenient starting point.

Figure 8.13 shows the results of this fit. Now, even at the 2σ level, a broad peak has been fitted in preference to the narrow gamma-ray peak. What is not obvious from the fit is that the peak at channel 1261 is also of the wide line shape; however, ROBFIT prefers to split any structure in this region into two narrow peaks.

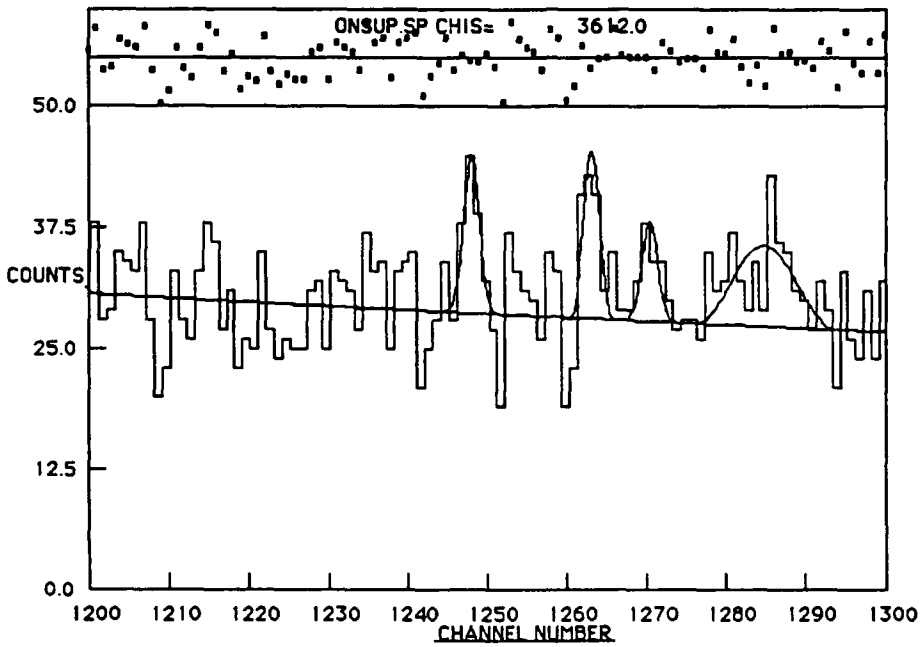


Figure 8.13 High supernova region after the fifth round of fitting. The code has started to find broad structures in this region.

We could now cycle the code a few times to home on the exact width of the peak, but we would like to throw another feature of the code into the analysis. If the broad peak at channel 1285 is actually a supernova peak, then we must also see a signal around the 870 channel region. This now allows us to illustrate another feature of ROBFIT; that is the ability to add peaks to the peak list. From the last run, we find the peaks within the 870 region have been fitted with the values given in Table 8.8.

Table 8.8 List of the peaks found in the 870 channel region in the fifth round of fitting.

```
OUTPUT FROM ROBF1T
IP=    3
 175 PEAKS 4040 CHAN, CHIS=       3612. NITB=   12
CUTOFF=   2.00
 1.0682  1.0000  1.3065   .9993  1.0065  1.0000

                    "other peaks"

 804.359  3.363     12.737    .862        69.68     46.75  2
 818.784   .588      2.564    .409        41.44     17.67  1
 848.065   .517      2.671    .397        49.88     17.04  1
 853.661   .196      2.947    .399       172.36     22.98  1
 863.647   .437     13.940   1.092       675.54     60.79  2
 872.587   .179      3.456    .392       235.61     28.84  3
 905.262   .681      2.754    .372        38.25     17.03  1

                    "other peaks"
```

It is necessary to introduce another peak, of the wide line standard, into this region. The precise starting values for areas and widths are not that important at this stage as the code will re-calculate these when it refits the region. However, we must give the peak some starting area or else the code will simply remove it. We must reduce the size of a nearby peak by the same amount because all the areas should sum to the same value as the previous fit. The new peak list is shown in Table 8.9. No errors need to be set at this point as they will be re-calculated during fitting.

Table 8.9 Modified list of peaks for input into round sixth of fitting.

```
OUTPUT FROM ROBF1T
IP=    3
 175 PEAKS 4040 CHAN, CHIS=       3612. NITB=   12
CUTOFF=   2.00
 1.0682  1.0000  1.3065   .9993  1.0065  1.0000

                    "other peaks"

 804.359  3.363     12.737    .862        69.68     46.75  2
 818.784   .588      2.564    .409        41.44     17.67  1
 848.065   .517      2.671    .397        49.88     17.04  1
 853.661   .196      2.947    .399       172.36     22.98  1
 863.647   .437     13.940   1.092       675.54     60.79  2
 872.587   .179      3.456    .392       135.61     28.84  1
 872.000  0          4        0          100         0     3
 905.262   .681      2.754    .372        38.25     17.03  1

                    "other peaks"
```

Step 6 is to run the output files of step five, plus the extra peak, back through the fitter. The starting values are:

4,12
3,3,1
2.61,0.36,927.2,2.47,0.28,1050.2
12.8,0.6,717,12.8,1.0,863
8.7,2,1260,8.7,2,1286

We have re-defined the width range of the fitted wide line standard according to the width of the peak fitted in step five. The results of this fit are illustrated in Figure 8.14 which shows the two supernova regions. From the figure we see that the code holds onto the wide line peak at the low supernova region. It also moves the peak position to channel 873.4 ± 1.1 with a width of 5.6 ± 2.0 channels and an area 116 ± 71. We also see some change at the high supernova region. The peak at channel 1264 now looks dramatically wider. This indicated that maybe the peak at channel 1270 is part of the same peak. This was also the case in our second data sample. We can perform a further cycle of the code with the 1270 peak removed.

The final step was to perform this re-run. The starting values were:

4,12
3,3,1
2.61,0.36,927.2,2.47,0.28,1050.2
12.8,0.6,717,12.8,1.0,863
5.6,2.0,873,10.4,2.0,1284.

where we have re-defined the width range for the wide line standard using the values output from the last fit. The final fits to the supernova regions are shown in Figure 8.15 and these are to be compared with the fits detailed in the supernova analysis[1]. The fitting presented in this section has been a complete re-analysis of the supernova data. It was somewhat reasssuring for us to be able to come up with the same answer in this second completely separate analysis!

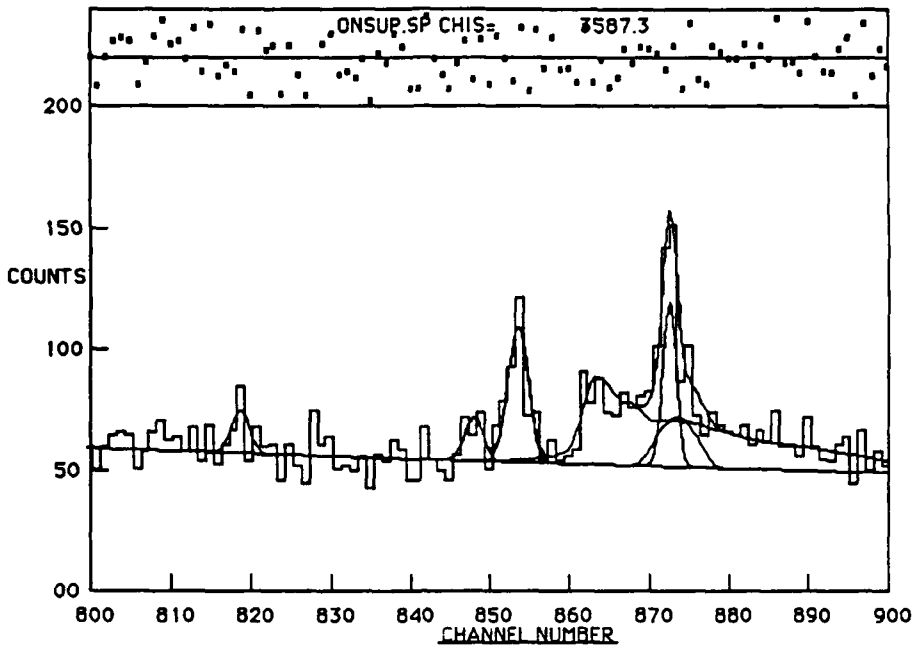


Figure 8.14a Low energy supernova region after sixth round fitting.

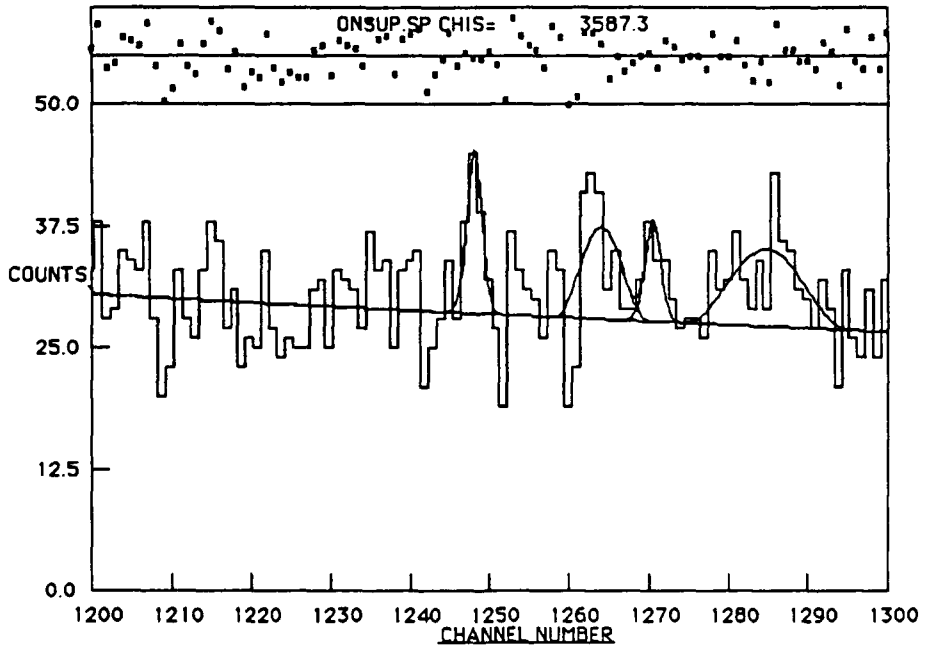


Figure 8.14b High energy supernova region after sixth round fitting.

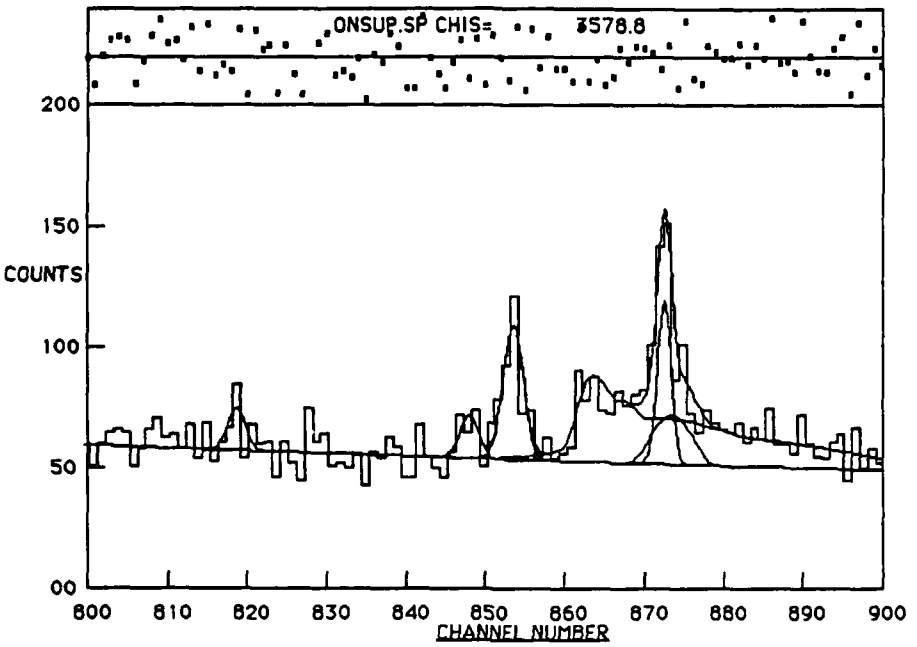


Figure 8.15a Result of the whole supernova analysis on the low energy region.

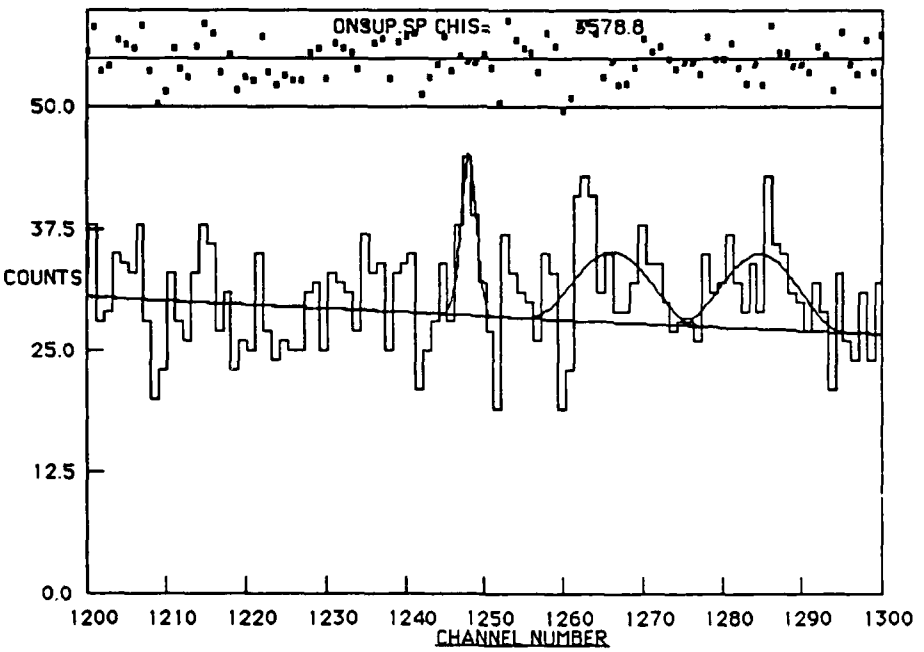


Figure 8.15b Results of the whole supernova analysis on the high energy region.

This concludes the analysis of the supernova data. We have pointed out the salient points of the analysis, and highlighted some of the features of ROBFIT that were used to extract the signal. We will complete this section by showing a list of peaks found during this re-analysis. This list, shown in Table 8.10, contains the real peaks and a number of spurious peaks which we have to live with. The identification of these spurious peaks is, as they say, left as an exercise for the reader.

Table 8.10 final peak list for the supernova analysis.

```
OUTPUT FROM ROBFIT
IP=    3
 184 PEAKS 4040 CHAN, CHIS=       3579. NITB=   12
CUTOFF=   3.00
 1.0682  1.0000  1.3065   .9993  1.0065  1.0000
  60.694  2.099      4.399   2.495       2897.44   2919.56  1
  62.952   .183      2.313    .638       3333.87   2749.00  1
  73.532   .217      3.513    .516       3655.14    463.66  1
  84.792   .211      2.299    .478       1104.03    204.94  1
 108.009  1.067      3.388    .729        363.94    195.20  1
 139.796   .066      2.266    .146       4844.12    284.06  1
 153.567  1.263      3.311    .551        194.14    125.82  1
 166.226   .802      3.309    .550        293.72    121.14  1
 177.388  1.195      3.185    .548        175.42    110.98  1
 189.385   .898      3.237    .547        225.76    106.54  1
 197.614  1.039      1.500   4.991        319.20    571.81  3
 200.631   .061      2.228    .150      10282.21    546.46  1
 243.772   .972      3.240    .539        152.18     78.17  1
 294.514  1.032      3.103    .532        101.48     57.52  1
 305.324   .825      3.308    .530        133.41     56.80  1
 330.201   .653      2.960    .527        127.57     48.31  1
 336.698   .800      3.052    .535        121.53     52.26  1
 340.727   .504      3.343    .559        219.19     56.20  1
 347.918   .989      3.107    .525         84.86     45.04  1
 352.694   .559      3.060    .530        143.17     45.61  1
 368.365   .826      3.058    .520         88.71     40.87  1
 376.466   .635      3.055    .519        111.68     40.35  1
 394.779   .840      3.130    .516         80.14     37.27  1
 404.166   .810      3.287    .514         88.19     37.13  1
 412.210   .717      2.908    .513         81.03     34.20  1
 439.814  1.041      3.051    .508         53.21     31.28  1
 451.496   .232      2.810    .506        224.86     37.05  1
 485.756   .210      2.365    .467        207.02     37.37  1
 514.597   .655      3.002    .544         76.73     29.52  1
 526.387   .057      3.636    .134       1807.44     63.07  1
 586.440   .387      3.908    .497        161.51     29.42  1
 600.108   .672      2.863    .483         61.61     23.66  1
 603.694   .736      2.743    .480         51.44     22.99  1
 615.727   .242     12.554    .315       1487.70     80.54  2
 620.838   .482      3.159    .476        109.76     33.94  1
 649.024   .217      2.926    .468        189.67     29.28  1
 662.622   .527      2.814    .459         62.68     20.83  1
```

685.426	.876	2.754	.452	35.34	19.03	1
704.788	.653	2.936	.446	50.74	19.97	1
716.921	.196	13.115	.660	1918.03	89.06	2
721.061	.953	2.819	.443	45.50	30.43	1
744.972	.494	2.839	.437	72.23	24.24	1
756.777	.572	2.749	.433	51.53	21.73	1
763.702	.762	2.776	.429	37.82	18.20	1
768.704	.634	2.923	.427	48.95	18.67	1
777.099	.221	3.337	.423	194.00	25.25	1
818.771	.551	2.583	.445	48.54	18.66	1
848.060	.539	2.675	.417	50.22	17.88	1
853.661	.206	2.960	.418	172.89	24.13	1
863.650	.489	14.140	1.275	663.67	70.32	2
872.431	.184	1.898	.613	137.64	62.62	1
873.362	1.066	5.620	2.036	115.65	70.71	3
905.170	.721	2.758	.390	37.75	18.13	1
911.068	.634	2.502	.396	47.73	20.09	1
913.896	.509	2.528	.406	60.04	20.51	1
927.227	.167	2.601	.389	181.22	24.12	1
932.253	1.234	12.729	1.286	207.51	46.34	2
942.508	.454	2.534	.544	84.95	27.77	1
973.298	.499	2.445	.334	41.52	14.67	1
998.759	.764	2.520	.363	29.84	15.98	1
1004.391	.761	2.551	.358	31.67	16.05	1
1013.233	.540	2.589	.351	46.26	16.82	1
1022.349	.210	2.458	.345	126.59	20.75	1
1050.172	.139	2.571	.280	186.12	20.24	1
1058.730	.572	2.434	.273	32.18	13.38	1
1076.485	.264	2.536	.261	85.16	16.13	1
1082.278	.348	2.585	.263	75.44	17.50	1
1086.373	1.363	12.934	1.506	189.80	42.48	2
1101.149	.372	2.370	.235	52.40	15.19	1
1135.903	.799	2.372	.190	20.84	12.27	1
1140.699	.639	2.375	.183	26.73	12.53	1
1145.531	.764	2.346	.175	21.84	12.15	1
1151.266	.501	2.298	.165	33.22	12.55	1
1157.214	.339	2.338	.154	52.48	13.49	1
1171.208	.674	2.318	.121	23.51	11.76	1
1197.519	.849	2.290	.047	23.62	15.06	1
1248.133	.389	2.229	.196	39.89	12.52	1
1266.239	1.591	10.528	2.004	77.19	22.63	3
1284.750	1.511	10.648	2.004	83.13	23.17	3
1317.989	.521	2.126	.298	24.65	10.72	1
1345.036	.483	2.119	.335	26.03	10.68	1
1418.470	.387	2.273	.430	36.49	11.36	1
1429.899	.442	1.789	.444	22.21	9.67	1
1438.962	.616	2.040	.456	18.23	9.70	1
1464.780	.779	2.052	.489	14.34	9.34	1
1515.628	.658	1.773	.836	20.94	14.07	1
1531.591	.579	1.681	.578	13.66	8.43	1
1566.985	.545	1.885	.627	17.36	9.09	1
1668.445	.295	1.500	1.108	39.86	17.70	1
1683.262	.353	1.500	.813	21.49	9.90	1
1693.667	.292	2.479	.647	50.44	12.14	1
1699.163	.415	1.506	.767	16.38	8.53	1
1735.214	.596	1.500	.979	13.09	9.65	1
1765.917	.775	1.500	.800	8.59	7.25	1
1770.018	.624	1.927	.799	14.04	8.38	1

1845.229	.366	2.522	.844	34.59	10.18	1
1866.651	.374	1.500	.998	17.31	8.61	1
1875.819	.589	2.489	1.053	24.42	11.30	1
1884.998	.727	1.919	.871	10.92	7.52	1
1889.057	.692	1.839	.875	10.63	7.38	1
1925.358	.332	1.500	.970	17.94	8.19	1
1932.320	.473	1.500	1.082	13.59	8.48	1
1947.043	.815	1.674	.908	8.10	6.97	1
1955.803	.556	1.500	1.109	14.04	9.61	1
1988.858	.619	1.500	1.073	11.22	8.33	1
2046.298	.463	1.500	1.038	11.90	7.20	1
2075.263	.411	1.638	.964	14.81	7.67	1
2094.686	.435	1.500	1.051	13.33	7.49	1
2123.083	.714	2.268	1.010	13.34	7.89	1
2144.213	.596	1.733	1.284	13.31	9.34	1
2176.391	.658	2.648	1.037	18.89	8.78	1
2187.397	.590	1.500	1.192	8.84	7.06	1
2201.455	.526	1.693	1.169	12.71	8.05	1
2239.400	.364	1.500	1.129	13.90	7.18	1
2255.712	.582	1.761	1.080	11.46	7.18	1
2283.425	.347	1.500	1.135	14.26	7.19	1
2286.899	.529	1.517	.944	11.13	7.09	1
2290.207	.485	1.865	1.238	14.60	8.07	1
2293.437	.300	1.500	1.504	17.97	9.19	1
2296.064	.437	1.507	.751	14.31	7.53	1
2301.805	.587	1.500	1.183	11.18	8.08	1
2306.096	.791	3.134	1.285	22.10	10.55	1
2336.860	.818	1.500	1.249	7.04	6.71	1
2366.724	.493	2.149	1.084	17.52	8.09	1
2384.186	.643	2.397	1.180	15.73	8.17	1
2390.088	.436	1.500	.754	12.51	6.86	1
2398.710	.537	1.500	1.362	10.63	8.25	1
2405.666	.513	1.500	1.346	11.40	8.87	1
2410.413	3.684	18.163	2.097	61.55	21.56	3
2418.778	.793	1.500	1.283	7.06	7.42	1
2459.491	.405	1.500	1.180	10.99	6.39	1
2468.469	.430	1.500	1.190	10.16	6.22	1
2631.797	.790	1.500	1.591	8.64	8.45	1
2653.740	.918	3.068	1.317	13.80	7.82	1
2663.193	.535	1.500	1.062	10.51	7.25	1
2667.097	.513	1.500	.897	11.25	7.32	1
2685.464	.471	1.500	1.877	13.84	10.03	1
2692.832	.751	1.500	1.427	9.07	8.40	1
2767.213	.427	1.500	.877	11.32	6.31	1
2773.594	.557	1.500	1.605	8.86	6.87	1
2860.479	.418	2.821	.919	27.58	8.29	1
2872.928	.395	1.500	.681	11.60	5.73	1
2883.908	.482	1.500	.841	9.23	5.51	1
2931.375	.380	1.500	1.243	10.86	6.06	1
2934.854	.814	2.737	1.451	11.79	6.80	1
2942.055	.849	2.305	1.386	8.22	5.82	1
2979.661	.739	2.778	1.515	14.26	7.58	1
2992.118	.392	1.500	.703	10.66	5.40	1
3011.691	.522	1.500	1.310	7.78	5.31	1
3035.132	.428	1.654	.859	10.38	5.42	1
3054.996	.589	1.500	.997	10.88	8.12	1
3072.808	.647	2.465	1.434	11.75	6.32	1
3088.633	.624	1.500	1.733	7.51	6.32	1

3125.981	.738	3.311	1.455	16.27	7.18	1
3138.504	.788	4.035	1.460	20.39	7.84	1
3169.787	.746	4.675	1.472	27.38	8.71	1
3216.957	.596	1.500	1.019	10.23	7.72	1
3269.666	.509	1.500	1.292	11.14	7.57	1
3334.729	.488	1.635	1.098	8.61	5.10	1
3362.401	.372	1.500	1.236	9.54	5.11	1
3376.493	.345	1.500	1.340	10.20	5.40	1
3384.263	.723	2.825	1.656	12.99	6.71	1
3388.465	1.025	2.351	1.633	6.62	5.34	1
3409.419	.554	1.847	1.324	8.19	5.10	1
3421.797	.697	1.502	1.406	5.22	4.51	1
3427.537	.770	3.583	1.575	16.22	6.98	1
3473.406	.465	1.500	1.619	7.11	4.80	1
3483.158	.578	1.500	1.084	5.96	4.41	1
3491.737	.403	1.500	.867	12.08	6.32	1
3528.078	.642	1.500	1.120	9.71	8.09	1
3552.519	.334	1.500	1.179	13.28	6.59	1
3565.152	.422	1.500	.788	8.42	4.64	1
3685.837	.689	1.500	1.298	5.04	4.29	1
3708.966	.930	1.500	1.554	3.57	4.05	1
3757.831	.413	1.500	.781	8.97	4.70	1
3823.580	.441	1.500	1.468	7.39	4.72	1
3832.792	1.017	3.841	1.701	12.34	6.30	1
3845.382	.747	2.852	1.640	11.16	5.89	1
3883.821	.604	1.500	1.157	7.97	6.10	1
3978.655	.838	2.270	1.747	6.51	4.81	1
4026.423	.490	1.500	1.786	6.02	4.37	1
4031.140	.409	1.500	.754	8.06	4.33	1
4064.213	.546	1.500	1.122	5.60	4.01	1
4073.579	.361	1.500	1.136	9.06	4.68	1

8.3 Solar seismology data

In this section we illustrate how ROBFIT can be used on data in which the underlying channel fluctuations do not follow Gaussian statistics. For gamma-ray spectra, the variation in an actual channel value is expected to have a standard deviation σ equal to f_i^2, as shown in Figure 8.16.

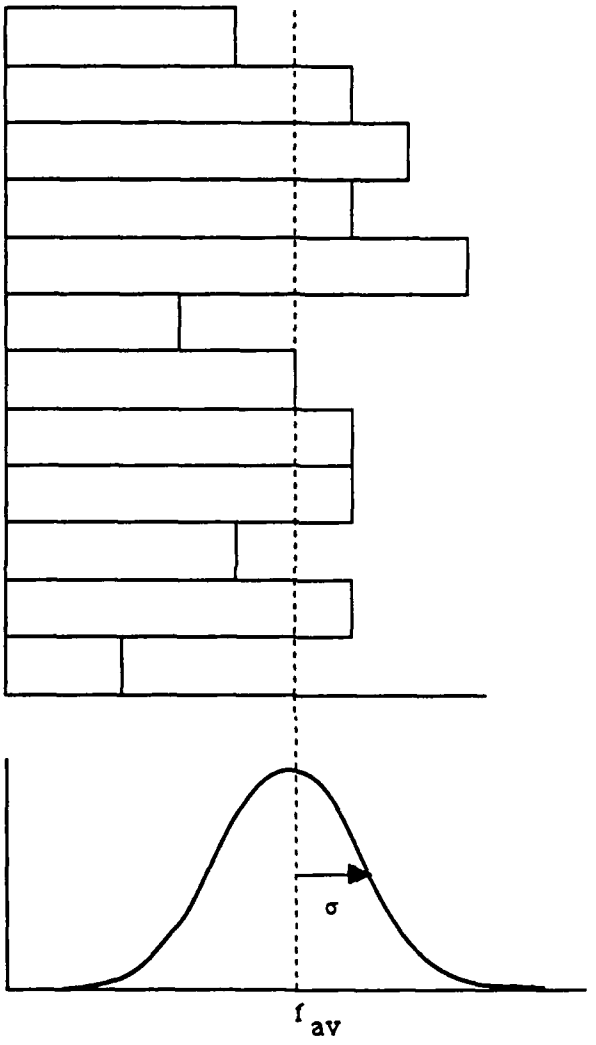


Figure 8.16 Illustrates the Gaussian fluctuation of the data points about the mean f_{av}.

ROBFIT uses this information in its minimization algorithm. The maximum likelihood technique assumes that the underlying statistical fluctuations are Gaussian. Using the terminology of Chapter 4, this can be seen by the following discussion. The probability of making an observation f_i, assuming a

Gaussian distribution within the channel of standard deviation σ_i is given by,

$$\text{Prob} = \frac{1}{\sigma_i\sqrt{2\pi}}\exp\left(-\frac{1}{2}\left[\frac{(f_i-f_{th})^2}{\sigma_i^2}\right]\right)$$

Thus, the probability of making a set of measurements i=1...N, is given by

$$\text{Probs} = \prod_i \frac{1}{\sigma_i\sqrt{2\pi}}\exp\left(-\frac{1}{2}\left[\frac{(f_i-f_{th})^2}{\sigma_i^2}\right]\right)$$

for a given fit to the data f_{th}. The maximum likelihood method attempts to maximize this probability by varying the coefficients of the fit. Maximizing this probability is equivalent to minimizing the sum in the exponential. In other words, minimizing the χ^2 of the fit is identical to the set of procedures outlined in Chapter 4.

With certain data; however, the underlying probability distribution does not follow Gaussian statistics. Now, the minimizing algorithm SMSQ is not operating on the correct probability function and is not obtaining the 'best fit' to the data anymore. In these situations, ROBFIT can still be used. Estimating the weights on each data point from fluctuations within the data, can offset some of the effects of using an incorrect probability distribution. This section is intended to give the user confidence in using ROBFIT.

Solar seismology is the study of the vibrational modes of the sun. Observations of surface motions results in a frequency spectrum which contains peaks at resonant frequencies[24]. Analysis of this data requires the unambigous determination of all these resonant modes. A more detailed view of some of these modes can be seen in Bamford[25]. These data contain channel to channel fluctuations that are distributed as χ^2 with two degrees of freedom, (see Duvall and Harvey[26,27]) and are non-Gaussian. Figure 8.17 illustrates this situation.

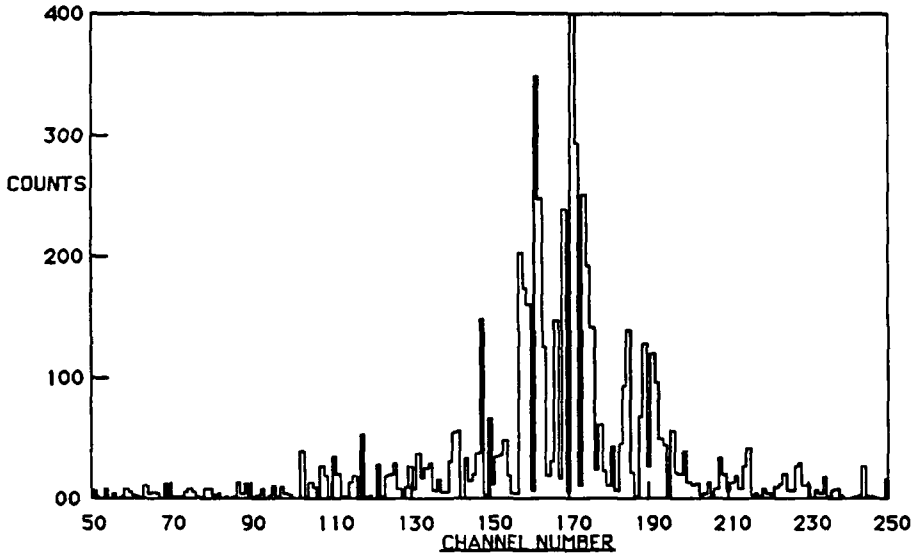


Figure 8.17 Illustration of the large channel to channel fluctuations in the solar spectrum. See figure 8.4b for comparison with Gaussian fluctuations.

The following tests use artificially generated data that contain channel to channel fluctuations calculated using the probability distribution of equation 8.1. This distribution mimics the actual Solar spectrum.

$$P(A) = \frac{1}{A_o} e - \frac{A}{A_o} \qquad \text{..... (8.1)}$$

The tests have been organized as follows. Twenty-five individual spectra have been generated, and the data fluctuations have been randomized according to the probability distribution of equation 8.1. Each of these spectra contain three Lorentzian shapped peaks with positions, widths, and stengths given by:

Position (channels)	Width (channels)	Strength (area)
167.66	22.22	6771
501.00	13.33	4188
834.33	8.90	2796

Twenty-five separate fits have been performed on this data. The results are shown in Table 8.11. The table contains the results of two methods of analysis. One uses individual channel weightings corresponding to $\omega_i = \frac{1}{\text{data value}}$; the

other estimates the weights from the fluctuation in the data. These weighting schemes have been described in Section 4.1.3. Table 8.11 also shows how accurately the code calculates the errors on these fits. From the 25 fits we can take each of the fitted values (position, width, and strength), and calculate a mean value. We can also deduce the standard deviation and error on the mean value. However, from each fit, we are given the estimated error on the values. This estimated error is effectively a measure of the standard deviation mentioned above. We can average these estimates, and if our error analysis is correct, the average should be close to the standard deviation calculated from the spread in the values.

The table shows that the code correctly reproduces the positions, widths, and strengths in both weighting cases. However, when the weights are taken as the inverse of the data values, as in gamma-ray spectroscopy, the errors calculated from the fit are slightly underestimated. Selecting the option for calculating the weights from the fluctuations in the data is seen to improve the error estimates. Of course, this is still an approximation to the weights that should be used in this analysis.

Table 8.11 Illustration of the effects of different weighting schemes on ROBFIT error analysis.

$$\text{Weights} = \frac{1}{\text{data value}}$$

Position and error	Standard Deviation	Estimated Standard Deviation
167.29 +/- 0.28	1.40	0.82
501.09 +/- 0.28	1.40	0.73
834.27 +/- 0.21	1.05	0.57
Width and error		
22.23 +/- 0.86	4.10	1.69
15.58 +/- 0.67	3.37	1.44
10.79 +/- 0.92	4.60	1.35
Strength and error		
7207 +/- 194	968	460
4177 +/- 123	613	285
2672 +/- 123	613	241

Weights = Calculated from data fluctuations.

Postition and Error	Standard Deviation	Estimated Standard Deviation
167.37 +/- 0.32	1.58	1.68
501.28 +/- 0.23	1.17	1.29
834.34 +/- 0.25	1.27	1.10
Width and error		
21.76 +/- 0.57	2.86	2.90
12.20 +/- 0.59	2.96	1.84
8.290 +/- 0.38	1.89	1.49
Strength and error		
6139 +/- 136	681	756
3980 +/- 187	936	587
2725 +/- 151	756	458

On closer inspection, the non-Gaussian probability function that is to be minimized is[26,27]

$$S = N\ln(f_{th}) + \frac{1}{f_{th}} \sum_{i}^{N} f_i$$

where the f_i's are the data points and f_{th} is the theoretical model described in Chapter 4, N is the total number of data points. But we can write an effective χ^2,

$$\chi^2 = \sum_{i} \frac{(f_i - f_{th})^2}{f_{th}^2}$$

which is in the same form as equation 4.1. Here, however, the weights are given by $\omega_i = \frac{1}{f_{th}^2}$. This function has a minimimum when,

$$\frac{\partial \chi^2}{\partial c} = - \sum_{i} \frac{2}{f_{th}^2} (f_i - f_{th}) \frac{\partial f_{th}}{\partial c} = 0$$

which is in the same form as the minimum of equation 8.2. Again, the minimum happens when $f_i = f_{th}$. The minimization has been reduced to a standard least-squares with the weights altered as suggested above. So, a correct analysis of this data would involve altering the weighting scheme *within the code so that it takes into consideration the* $\frac{1}{f_{th}^2}$ dependence while minimizing. However, for the present, we wish to only illustrate the versatility of ROBFIT by showing that a good fit can be achieved with no modifications at all.

9 Conditions in which the authors would use other standard codes

In this chapter we will outline some of the situations in which other spectral analysis codes may perform better than ROBFIT.

1) The first case is one in which the peak shapes and background for the spectrum are known analytically. The ROBFIT code builds up the peak shapes from back-to-back cubic splines and the background from cubic splines. These functions are approximations which introduce their own errors into the fits. Normally, the error introduced by the splines is very small compared to the statistical errors in the spectrum; however, they are always present. A code with the true shape and background built into it from the outset would have an easier time locating peaks within the spectrum. This assumes, of course, that the code can also correctly deal with complex backgrounds and overlapping peak regions[4,5,28,29].

2) A problem can arise when the spectrum contains very large peaks. The ROBFIT code assumes that each peak in the spectrum is exactly the same shape as the standard peak of its type. In gamma-ray spectra, the real peak shape has a high energy leading edge, and low energy tailing. In most cases, these inaccuracies, again caused by the peak shape approximation, do not concern the user as they are masked by the errors introduced by the data fluctuations. However, in the case of a very large peak, where the statistics are better, there is a possibility that they will show up. The problem is identified by braiding of the large peak as the code tries to correct for the inaccuracy by adding more peaks. The user must be careful and not confuse this with braiding caused by incorrect peak width specification.

This problem can be reduced by using different standard peaks in different energy regions, and by allowing more and more constants in the background. This is recommended if the object is to find small peaks in the vicinity of the large peaks. If, however, the object is to find properties involving the large peak, then there is no substitute for analytical information. Furthermore, in these situations, the background is relatively unimportant and the overlapping of peaks is probably negligible. In this case

there is no reason to fit the entire spectrum, which is the principle advantage of ROBFIT.

3) In order for the code to pick out peaks that are in the noise, ROBFIT must have an accurate representation of the background across the peak region. ROBFIT uses the entire spectrum to define the background function, and for purposes of speed, compresses the spectrum by a factor of 16. This is normally not a problem since the background knots should be further apart than the full width at half maximum of the peaks. However, if one tries to fit a spectrum of only 40 channels, then no more than 2 constants can be defined in the background, and these rather poorly. Numerous other codes exist for treating short segments of spectra. The authors recommend using at least 32 channels per background constant, but if this is not possible, then ROBFIT is probably not the best code to use to analyze the data.

References

1. A.C. Rester, R.L. Coldwell, F.E. Dunnam, G. Eichhorn, J.I. Trombka, R. Starr, and G.P Lasche Ap.J. vol. 342, pp. L71-3, July 15, 1989.

2. G.F. Knoll, *Radiation Detection and Measurement*, New York, John Wiley and Sons, (1989).

3. P. Quittner, *Gamma-ray Spectroscopy*, London, Adam Hilger (1973) p. 67.

4. G.W. Phillips and K.W. Marlow, Nucl. Instr. and Meth. **137** (1976) 525-536.

5. M. Hillman, Nucl. Instr. and Meth. **135** (1976) 363-368.

6. L. Kokta, Nucl. Instr. and Meth. **112** (1973) 245-251.

7. D. D. Burgess, Nucl. Instr. and Meth. **221** (1984) 593-599.

8. M.A. Mariscotti, Nucl. Instr. and Meth. **50** (1967) 309-320.

9. A. Robertson, W.V. Prestwich and T.J. Kennett, Nucl. Instr. and Meth. **100** (1972) 317-324.

10. R.G. Helmer and M.A. Lee, Nucl. Instr. and Meth. **178** (1980) 499-512.

11. P.R. Bevington, *Data Reduction and Error Analysis for the Physical Sciences*, New York, McGraw-Hill, 1969.

12. R.L. Coldwell, Nucl. Instr. and Meth. **A242** (1986) 455.

13. R.L. Coldwell, *Radiative Properties of Hot Dense Matter*, World Scientific, 1983, pp 315-349.

14. I.J. Schoenberg, Quant. Appl. Math. **4** (1946) 45-99 p.67.

15. J.C. Holladay, Math. Tables Aids Computation, **11** (1957), 233-243.

16. J.H. Ahlberg, E.N. Nilson and J.L. Walsh, *The Theory of Splines and their Applications*. Academic Press (1967) p.3.

17. C. de Boor, *A Practical Guide to Splines*. Springer Verlag (1978) p.180.

18. S. Baker and R.D Cousins, Nucl. Instr. and Meth. **221** (1984) 437-442.

19. G.W Phillips, Nucl. Instr. and Meth. **153** (1978) 449-455.

20. W.H. Press, B.P. Flannery, S.A. Teukolsky and W.T. Vetterling, *Numerical Recipes*, New York, Cambridge Univer.ity Press, 1987.

21. W.T. Morton, Nucl. Instr. and Meth. **A272** (1988) 861-865.

22. D. Mihalas, *Stellar Atmospheres*, New York, Freeman, (1979).

23. W.R. Falk, Nucl. Instr. and Meth. **220** (1984) 473-478.

24. J. Christensen-Dalsgaard, *Advances in helio-and asterioseismology*, I.A.U. Symposium (123rd, 1986, Arhus Denmark), pp 3-18.

25. G.J. Bamford, H.B. van der Raay, P.L. Palle and T. Roca Cortes, Proc. of the Fourth European Meeting on Solar Physics, 1-3 Oct. 1984 Noordwijkerhout, the Netherlands.

26. T.L. Duvall, J.W. Harvey and M.A. Pomerantz, *Advances in helio-and asterioseismology*, I.A.U. Symposium (123rd, 1986, Arhus Denmark), pp 37-40.

27. E.R. Anderson, T.L. Duvall and S.M. Jefferies, Modeling of Solar Oscillation Power Spectra. N.O.A.O. Tucson Arizona. To be published.

28. R.B. Welch, F. Gyger, D.T. Jost, H.R. von Gunten and U. Krahenbuhl, Nucl. Instr. and Meth. A269 (1988) 615-622.

29. H. Mochner, Nucl. Instr. and Meth. A258 (1987) 246-249.

30. T.J. Kennett, W.V. Prestwich and R.J. Tervo, Nucl. Instr. and Meth. 216 (1983) 205-218.

31. D.D. Burgess and R.T. Tervo, Nucl. Instr. and Meth. 214 (1983) 431-434.

32. G. Winter, Nucl. Instr. and Meth. A258 (1987) 119-126.

33. W. Westmeier, Nucl. Instr. and Meth. 180 (1981) 205-210.

Appendix A

ROBFIT code listing

```
C*************************************************************************
      PROGRAM ROBFIT
C*************************************************************************
        IMPLICIT REAL*8 (A-H,O-Z)
        INTEGER*2 I35
C *** SCONS IS A COMMON CONTAINING THE BACKGROUND CONSTANTS
         COMMON/SCONS/DU(150),XPS(5),XPL(5),IP,NGPCAL
       REAL*4 F,FA,FB,AMAX1,AMIN1,SQRT,CUTOFF,WX,FT,SU,ALR,XE,CHIOLD
     # ,AMDIST,XOFF
        CHARACTER*64 NAGR,NAPK,NABKG,NAOBKG,NAOPK,NADAT,NAWT,NADIR,NA
        CHARACTER*4 IFW,NOGP,IFIXW,IRBKG,IVFW
        CHARACTER*1 ANS,APH
        CHARACTER*4 HDAT,IDAT,Z4DAT,BKGF,EXTWT
        DIMENSION F(8192),FB(8192),WX(8192),FA(8192)
        DIMENSION C(256 ),W(256 ),XP(256 ),PD(768 ),IPT(256 )
        DIMENSION BCONS(100),ARAT(5),WRAT(5)
        DIMENSION IHD(8192),SC(256 ),SW(256 ),SXP(256 ),WF(256 )
        DIMENSION CHIOLD(3),ITIM(2),FW(3,5),SFW(5,5)
C *** START OF SETWIN
        INTEGER*4 TOOLBX,IFRWIN,PTR,FRONTWINDOW,SIZEWINDOW
        INTEGER*2 IRECT(4)
        INTEGER*1 IGREC(154)
        PARAMETER (NEWWINDOW=Z'91320000',PTR=Z'C0000000',
     # FRONTWINDOW=Z'92480000',SIZEWINDOW=Z'91D112C0',
     # MOVEWINDOW=Z'91B112C0')
        DATA XVB,XVE/2*0.D0/,FA,FB/ 16384*0./,NV/0/
        DATA IPW/0/,PCON/1.D6/,BWSPW/-9.D9/
        DATA ALPHA/1.D-12/BCONS/100*0.D0/
        DATA NE/0/,ICONT/0/,FA/8192*0./
        DATA IRECT/0,0,450,700/,IGREC/154*0/
        ITWIN=TOOLBX(FRONTWINDOW)
       CALL TOOLBX(SIZEWINDOW,ITWIN,600,450)
C *** END OF SETWIN *******

C *** READ IN THE STARTING VALUES FOR THE FIT. THIS FILE MUST BE OF THE FORM FILE
       PRINT*,' ENTER THE DIRECTION FILE NAME'
       IDIR=0
       READ(*,'(A)')NADIR
         IT=INDEX(NADIR,'.')
         IF(NADIR(IT+1 :IT+3).EQ.'INP')GOTO 3
       IDIR=1
       OPEN(13,FILE=NADIR)
2      READ(13,'(A)',END=950)NADIR
       IF(NADIR.EQ.'\'.OR.NADIR.EQ.'STOP'.OR.NADIR.EQ.'END')GOTO 950
```

```
3     OPEN(2,FILE=NADIR)
        NCALLB=0
        NGPCAL=0
        XT=0
C *** INITIALIZING THE STANDARD BY CALLING POLYG
        XTT=POLYG(XT,1,ST)
        DO 5 I=1,3
5       CHIOLD(I)=1.E30
        IREFIT=0
        IREC4=2
        PRINT*,' ENTER THE MAX NUMBER OF PEAKS'
        READ(2,*)NPPMAX
        PRINT*,NPPMAX
        NPPMAX=MAX0(1,MIN0(NPPMAX,255))
        WMAX=0
        DO 7 I=1,IP
        PRINT*,' ENTER W1,EW1,NC1,W2,EW2,NC2'
        READ(2,*)W1,EW1,NC1,W2,EW2,NC2
        WMAX=DMAX1(WMAX,W1+EW1,W2+EW2)
        PRINT*,W1,EW1,NC1,W2,EW2,NC2
        FW(2,I)=(W2**2-W1**2)/(NC2-NC1)
        FW(1,I)=W1**2-NC1*FW(2,I)
        EW12=(2*W1*EW1)**2
        EW22=(2*W2*EW2)**2
        SFW(2,I)=(EW22 -EW12)/(NC2-NC1)
        SFW(1,I)=EW12-NC1*SFW(2,I)
      PRINT*,' FW',I,FW(1,I),FW(2,I)
        DO 6 J=3,5
6       SFW(J,I)=0
      FW(3,I)=0
7     CONTINUE
        PRINT*,' ENTER CUTA, CUTB, NUMBER OF INTERVALS'
        READ(2,*)CUTA,CUTB,IMREF
        PRINT*,' CUTA,CUTB,IMREF ',CUTA,CUTB,IMREF
        CUTOFF=CUTA
        XVB=0
        XVE=1
        PRINT*,' ENTER FWID, VWID, OR VFWI, FOLLOWED BY NOBF OR NONK'
        READ(2,105)IFW,BKGF
        READ(2,*)NBMIN,NBMAX
        PRINT*,IFW,BKGF,NBMAX
        PRINT*,' ENTER GP ,F SP, OR NONE'
        READ(2,105)NOGP
        PRINT*,NOGP
105     FORMAT(10(A4,1X))
      IF(IFW.EQ.'FWID')WRITE(*,102)
102     FORMAT(' THE FIXED WIDTH OPTION IS IN EFFECT')
      READ(2,'(A)')NADAT
      READ(2,*)N1,N2
      PRINT*,' THE DATA IS FROM ',NADAT,' CHANNELS',N1,N2
      READ(2,'(A)')NAWT
      PRINT*,' THE WEIGHTS ARE FROM ',NAWT
      PRINT*,' BEFORE BREAD'
      EXTWT='YES '
      IF(NAWT.EQ.'NORM')EXTWT='NORM'
        CALL BREAD(XOFF,F,IHD,IHD,WX,N,EXTWT,NADAT,NAWT,N1,N2)
      PRINT*,' AFTER BREAD'
        READ(2,'(A)')NABKG
```

```
      PRINT*,' THE BACKGROUND CONS WILL BE PUT INTO ',NABKG
      READ(2,'(A)')NAOBKG
      PRINT*,' THE STARTING ESTIMATES FOR THE BKG WERE IN ',NAOBKG
      READ(2,'(1X,A1)')ANS
      LFLAG=0
      IF(ANS.EQ.'O')LFLAG=1
        READ(2,'(A)')NAPK
      PRINT*,' THE PEAKS FOUND WILL BE PUT INTO ',NAPK
      READ(2,'(A)')NAOPK
      PRINT*,' A STARTING ESTIMATE FOR THE PEAKS CAME FROM ',NAOPK
      PRINT*,' ARE THE PEAKS THE CORRECT HEIGHT?'
      READ(2,'(A1)')APH
        PRINT*,' ENTER THE FILE NAME FOR THE GRAPHICAL DATA'
        READ(2,'(A)')NAGR
      PRINT*,' THE GRAPHICAL DATA IS IN ',NAGR
        NPP=0
        CALL ROPKS(C,SC,W,IPT,WF,SW,XP,SXP,N,NPP,FW,XOFF,NAOPK)
        IF(NPP.EQ.0)GOTO 90
        IF(APH.EQ.'Y')GOTO 77
      BKGCUT=CUTA
      DO 65 I=1,50
        CALL BKGFIT(XOFF,F,FA,WX,N,BCONS,NV,NBMAX,BKGF,LFLAG,NCALLB,
     # BKGCUT,NAOBKG)
      IF(NCALLB.EQ.30)GOTO 65
      IF(NV.GT.NBMIN)GOTO 68
65    CONTINUE
68    CONTINUE
        DO 75 I=1,NPP
        C(I)=CFEST(W(I),IPT(I),F,FA,WX,XP(I)-XOFF,N,ERRSQ,IFW)
75      CONTINUE
      PRINT'(A/(3G15.6,I3))',' AFTER CFEST C W XP,IPT',
     # (C(I),W(I),XP(I),IPT(I),I=1,NPP)
77      CONTINUE
        CHIT=0
        DO 80 I=1,N
        I=I+XOFF
        FB(I)=POLYA(XI,C,XP,W,IPT,NPP)
        F(I)=F(I)-FA(I)
        CHIT=CHIT+(F(I)-FB(I))**2*WX(I)
80      CONTINUE
90      CONTINUE
        WRITE(*,99)W1,W2
99      FORMAT(' FOR THE FIRST 50 PEAKS, PEAK WIDTHS ARE ASSUMED'/
     # ' LINEAR BEGINNING AT',F7.2,' AND ENDING AT',F7.2)
        WRITE(*,101)N,NPPMAX,XOFF
101     FORMAT(I5,' CHANNELS, AT MOST',I5,' PEAKS.'/
     # '  ZEROTH CHANNEL IS ',F8.0)
        NIT=0
        IWF=2
        IF(IFW.EQ.'FWID')THEN
        PCON=1.D6
        IWF=4
        ENDIF
        IF(IFW.EQ.'FIXD')IWF=9
        IF(IFW.EQ.'FIXN')IWF=-9
290     CONTINUE
        WRITE(*,107)CUTOFF
107     FORMAT(' THE CODE WILL ATTEMPT TO ADD PEAKS UNTIL THE LARGEST',
```

```
     # ' RESIDUAL IS <',F10.2)
        XT=XOFF+N/2
        IF(NPP.GT.20)XT=.5*(XP(1)+XP(NPP))
        DO 295 I=1,N
       XI=I+XOFF
       FB(I)=POLYA(XI,C,XP,W,IPT,NPP)
        F(I)=F(I)+FA(I)-FB(I)
        WX(I)=WX(I)/(1+.5*FB(I)*FB(I)*WX(I))
295     CONTINUE
297    BKGCUT=CUTOFF
       IF(NV.GE.NBMAX.AND.BKGF.EQ.'CONT')BKGF='NONK'
       IF(NV.GE.NBMAX.AND.BKGF.EQ.'FIXK')BKGF='FINK'
       PRINT*,' NV,NBMAX,BKGF',NV,NBMAX,BKGF
        CALL BKGFIT(XOFF,F,FA,WX,N,BCONS,NV,NBMAX,BKGF,LFLAG,NCALLB,
     # BKGCUT,NAOBKG)
       IF(NV.LT.NBMIN)THEN
       BKGF='CONT'
       GOTO 297
       ENDIF
300     CHIS=0.D0
        DO 1119 I=1,N
        F(I)=F(I)-FA(I)+FB(I)
        WX(I)=WX(I)/(1-.5*FB(I)*FB(I)*WX(I))
       IF(EXTWT.NE.'YES ')WX(I)=1./AMAX1(1.,FA(I)+FB(I))
        CHIS=CHIS+(F(I)-FB(I))**2*WX(I)
1119    CONTINUE
        WRITE(*,1120)NV,CHIS
1120    FORMAT(' AFTER BKGFIT WITH',I5,' COEFS, CHIS=',E20.12)

C *** NCALLB = 30 IS A FLAG TO SAY THAT NO NEW FITTING WAS DONE

       IF(NCALLB.EQ.30)GOTO 290
       IPOSPK=0
       IF(IABS(IWF).EQ.9)GOTO 315
        CHIAVE=CHIOLD(1)
        DO 310 I=1,2
        CHIAVE=CHIAVE+CHIOLD(I+1)
310     CHIOLD(I)=CHIOLD(I+1)
        CHIOLD(3)=CHIS
        IF(CHIAVE-3*CHIS.LT.300)GOTO 315
        IF(IREC4.GT.0)GOTO 390

C *** REFIT SECTION

315     IREC4=2
       IF(NPP.EQ.0)GOTO 550
        IF(NPP.GT.15.AND.IFW.NE.'FWID'.AND.IABS(IWF).NE.9)
     # IWF=3
        IF(IFW.EQ.'VFWI'.AND.NPP.GT.20)IWF=4
        WRITE(*,1399)
1399    FORMAT(' REFITTING ALL THE PEAKS')
        IP1=1
        I=1
320     ILR=I
        KWF=5
        IF(IWF.EQ.1)KWF=6
        IF(IWF.EQ.4)KWF=7
        IF(IWF.EQ.9)KWF=9
```

```
      IF(IWF.EQ.-9)KWF=-9
        CALL PFIT(N,ILR,F,FB,WX,C,SC,W,IPT,WF,SW,XP,SXP,NPP,
     # CHIS,2,KWF,IP1,IPJ,XVB,XVE,FW,SFW,XOFF,IPW,PCON,WP)
        DO 330 K=IP1,IPJ
1857    FORMAT(I5,F8.2,F6.2,F7.1,F6.1,F6.2,F6.2,I2,F10.0)
330     WRITE(*,1857)K,XP(K),SXP(K),C(K),SC(K),W(K),SW(K),IPT(K),CHIS
        I=IPJ+1
        IF(I.LE.NPP)GOTO 320
        DO 340 I=1,3
340     CHIOLD(I)=1.E32
        WRITE(*,1832)CHIS
1832    FORMAT(' THE CHIS AFTER FIXING THE WIDTHS IS',E12.6)
        GOTO 550

C *** NORMAL PEAK ADDING SECTION


390     CONTINUE
        IQM=0
501     CONTINUE
        IF(NPP.GT.NPPMAX)GOTO 290
        NRESP=5*WMAX
        CALL RESL(F,FB,WX,FW,ALR,IQM,ILR,IFTC,NRESP,N,XOFF,
     # CUTOFF)
      IF(ALR.GT.0)IPOSPK=IPOSPK+1
      IF(IFTC.EQ.1.AND.IPOSPK.EQ.0)GOTO 315
        IF(IFTC.EQ.1.AND.IPOSPK.LT.3)IREC4=IREC4-1
        IF(IFTC.EQ.1)GOTO 290
        NPPO=NPP
        CHISO=CHIS
      IFITT=1
      IF(ALR.LT.0)IFITT=-1
        CALL PFIT(N,ILR,F,FB,WX,C,SC,W,IPT,WF,SW,XP,SXP,NPP,
     # CHIS,IFITT,IWF,I1,JP,XVB,XVE,FW,SFW,XOFF,IPW,PCON,WP)
        TNPKS=.1*(ALR-CUTOFF)**2
        TNPKS=DMAX1(TNPKS,1.D0)
        XT=ILR+XOFF
        WRITE(*,119)NPP,IQM,IWF,XT,WP,ALR,CHIS
119     FORMAT(I5,I3,I2,4G14.6)
      IF(ALR.LT.0.D0)GOTO 501
        IF(IABS(IWF).EQ.9)GOTO 501
        IF(NPP.GT.NPPO)GOTO 501
        IF(CHISO-CHIS.GT.TNPKS)GOTO 501
        CALL PFIT(N,ILR,F,FB,WX,C,SC,W,IPT,WF,SW,XP,SXP,NPP,
     # CHIS,0,IWF,I1,JP,XVB,XVE,FW,SFW,XOFF,IPW,PCON,WP)
        WRITE(*,119)NPP,IQM,IWF,XT,WP,ALR,CHIS
        GOTO 501

C *** NORMAL EXITING

550     CONTINUE
        CHIS=0
        DO 600 I=1,N
        CHIS=CHIS+(F(I)-FB(I))**2*WX(I)
        F(I)=F(I)+FA(I)
600     FB(I)=FB(I)+FA(I)
        WRITE(*,108)NPP,CHIS
108     FORMAT(' AT END OF MINIMIZATION',I4,' PEAKS, CHIS',E20.6)
```

```
      OPEN(UNIT=2,FILE=NABKG)
      WRITE(2,*)LFLAG
      WRITE(2,'(G20.12)')(BCONS(I),I=1,NV)
      CLOSE(2)
C *** OPEN THE GRAPHICS OUTPUT FILE
            PRINT*,'HELLO JUST OPENNED',NAGR
            OPEN(UNIT=3,FILE=NAGR,FORM='FORMATTED')
            WRITE(3,'(A,G20.12)')NADAT(1:8),CHIS
            WRITE(3,'(2I5)')IP
            OPEN(UNIT=19,FILE='TEMP')
            IPP=IP+1
            DO 620 I=1,IPP
            WRITE(19,'(A)')NA
620         WRITE(3,'(A)')NA
      CLOSE(19)
C *** WRITE THE RAW DATA PARAMETERS FOR FIT
      WRITE(3,'(A4/(A64))')EXTWT,NADAT,NAWT,NABKG,NAPK
      WRITE(3,'(2I5)')N1,N2
      WRITE(3,'(4G20.12)')(FW(1,I),FW(2,I),I=1,IP)
        CLOSE(3)
        WRITE(*,110)(XP(I),SXP(I),C(I),SC(I),W(I),SW(I),I=1,NPP)
110     FORMAT(F10.3,F7.3,F10.3,F7.3,F8.3,F6.3)
        OPEN(4,FILE=NAPK,STATUS='NEW')
        WRITE(4,'('' OUTPUT FROM ROBFIT''/'' IP='',I5)') IP
            WRITE(4,113)NPP,N,CHIS,NV,CUTOFF
            WRITE(*,113)NPP,N,CHIS,NV,CUTOFF
113         FORMAT(I5,' PEAKS',I5,' CHAN, CHIS=',F12.0,' NITB=',I5/
     # ' CUTOFF=',F7.2)
C *** THE BACKGROUND ERROR IS TAKEN TO BE THAT IN A ONE CONSTANT FIT
C *** TO ALL POINTS WITHIN THE RANGE OF ITS SPLINES ABOUT THE PEAK CENTER
C *** THUS THIS SOURCE OF ERROR IN C(I)**2 IS SQRT(SUM((FB-F))**2*WX/
C *** (SUM WX)))
        DO 800 I=1,NPP
        ITYPE=IABS(IPT(I))
        IRES=XP(I)-XOFF
        NBEG=IRES+W(I)*XPS(ITYPE)*4
        NBEG=MAX0(1,NBEG)
        NEND=IRES+W(I)*XPL(ITYPE)*4
        IF(NEND.LE.N)GOTO 710
        NEND=N
        NBEG=N-IRES
710     ANUM=0
        ADEN=0
        DO 720 J=NBEG,NEND
        ANUM=ANUM+(F(J)-FB(J))**2*WX(J)
720     ADEN=ADEN+WX(J)
      EB2=ANUM*NV/(ADEN*N)
C *** IN THIS VERSION OF THE CODE SC IS THE ERROR IN C**2*W
C *** AND IS CALCULATED IN QMIN
C *** THOUGH CHANGED HERE, NOTE THAT SC IS NOT USED ANYWHERE
C *** AND WILL BE RECALCULATED IN THE NEXT REFIT
800     SC(I)=DSQRT(SC(I)*SC(I)+EB2*W(I))
            CALL FWHM(NPP,ARAT,WRAT)
            WRITE(4,'(10F8.4)')(ARAT(I),WRAT(I),I=1,IP)
            DO 910 I=1,NPP
        ITEST=IABS(IPT(I))
        STR=C(I)*C(I)*W(I)
        STR=ARAT(ITEST)*STR
```

```
        SC(I)=ARAT(ITEST)*SC(I)
        SXP(I)=DMIN1(SXP(I),99.999D0)
        WOUT=WRAT(ITEST)*W(I)
        STTW=WRAT(ITEST)*SW(I)
        SC(I)=DMIN1(SC(I),999999.D0)
      STROUT=STR
      IF(IPT(I).LT.0)STROUT=-STR
            WRITE(*,114)XP(I),SXP(I),WOUT,STTW,STROUT,SC(I),IPT(I)
114         FORMAT(F9.3,F7.3,F11.3,F8.3,F14.2,F10.2,I3)
            WRITE(4,114)XP(I),SXP(I),WOUT,STTW,STROUT,SC(I),IPT(I)
910     CONTINUE
        CLOSE(4)
      IF(NOGP.EQ.'F SP')CALL FSPECT(1,N,FA,F,WX,C,XP,W,IPT,PD,
     # NPP,XOFF)
      IF(NOGP.EQ.'F PS')CALL PSPECT(1,N,FA,F,WX,C,XP,W,IPT,PD,
     # NPP,XOFF)
            IF(NOGP.NE.'GP  ')GOTO 920
C *** NOTE THAT PD IS USED BY RCPLOT FOR DERIVATIVE VALUES
920         CONTINUE

C *** RECALCULATING CUTOFF AND EITHER STOPPING OR CONTINUING
      IF(DABS(CUTA-CUTB).LT..1D0)GOTO 945
      IF(CUTOFF-CUTB.LT..1D0)GOTO 945
        CUTOFF=CUTOFF-(CUTA-CUTB)/IMREF
        IF(CUTOFF.GE.CUTB)THEN
        DO 940 I=1,N
        F(I)=F(I)-FA(I)
940     FB(I)=FB(I)-FA(I)
        GOTO 290
        ENDIF
945   IF(IDIR.EQ.1)GOTO 2
950     STOP
        END

C*****************************
C *** END OF THE MAIN ROUTINE ********************************************
C*****************************
C*************************************************************************
        SUBROUTINE PFIT(N,ILR,F,FB,WX,C,SC,W,IPT,WF,SW,XP,SXP,NPP,
     # CHIS,IFTC,IWF,JB,NPB,XVB,XVE,FW,SFW,XOFF,IPW,PCON,WPNNT)
C*************************************************************************
        IMPLICIT REAL*8 (A-H,O-Z)
            REAL*4 F,FB,AMAX1,AMIN1,SQRT,WX,FT,XOFF
        COMMON/SCONS/DUM(150),XPS(5),XPL(5),NTYPE
        DIMENSION F(1),FB(1),WX(1),C(1),SC(1),XP(1),SXP(1),W(1),
     # IPT(1),WF(1),SW(1),ECN(10),EXN(10),EWN(10)
     # ,CN(10),XN(10),WN(10),IPTN(10),WFN(10),FW(3,5),SFW(5,5)
C *** THIS ROUTINE REFITS THE PEAKS CLOSE TO ILR,  IF NO PEAK IS
C *** CLOSE TO ILR OR IF IFTC=0 A NEW PEAK IS ADDED
C *** IFTC=2 IMPLIES A REFIT, IN WHICH CASE ILR IS THE PEAK BEING
C *** REFITTED
C *** IFTC=-1 IMPLIES A NEGATIVE RESIDUAL IN WHICH CASE NO PEAK IS ADDED
C *** FINDING BEGINNING AND ENDING FIT POINTS WHEN REFITTING
        NPB=JB
        XT=ILR+XOFF
      IF(IFTC.LT.0)THEN
      XTS=XT
      XTL=XT
```

```
      DO 100 J=1,NTYPE
        WPG=DSQRT(DMAX1(3D0,FW(1,J)+XT*(FW(2,J)+XT*FW(3,J))))
      XTS=DMIN1(XTS,XT+XPS(J)*WPG)
      XTL=DMAX1(XTL,XT+XPL(J)*WPG)
100   CONTINUE
      GOTO 400
      ENDIF
        IF(IFTC.NE.2)GOTO 200
        INNT=IABS(IPT(ILR))
        XT=XP(ILR)
        XTS=XT+XPS(INNT)*W(ILR)
        XTL=XT+XPL(INNT)*W(ILR)
        GOTO 400
200     CONTINUE

C *** FINDING BEGINNING AND ENDING POINTS WHEN NO PEAK IS AT ILR
C *** WHICH IS NOW THE CHANNEL NUMBER WITH THE LARGEST RESIDUAL

C *** THIS IS WHERE WE FIND THE PEAK TYPE WITH THE MINIMUM INITIAL ERROR
        ERRC=1.D32
        DO 300 J=1,NTYPE
        XTT=ILR+XOFF
        CT=CQMIN(FB,F,WX,XOFF,N,XTT,WPG,J,FW,SFW,ERRSQ)
        IF(ERRSQ.GT.ERRC)GOTO 300
      CNNT=CT
        ERRC=ERRSQ
        INNT=J
      WPNNT=WPG
        XT=DMAX1(1.D0+XOFF,DMIN1(1.D0*N+XOFF,XTT))
300     CONTINUE
        XTS=XT+WPNNT*XPS(INNT)
        XTL=XT+WPNNT*XPL(INNT)

C *** NOW USING XTS AND XTL FIND PEAKS IN THE REGION OF INTEREST

400     CONTINUE
        CALL LOCATE(XTS,XP,NPP,JB)
        JB=JB+1
        CALL LOCATE(XTL,XP,NPP,JE)

C *** INCLUDING ALL POSSIBLE OVERLAPPING PEAKS BEWARE TIME GOES AS NP**3

405   JBM=JB-1
      DO 408 I=1,JBM
      ITYPE=IABS(IPT(I))
      IF(XTS.GT.XP(I)+W(I)*XPS(ITYPE))GOTO 408
      JB=I
      XTS=DMIN1(XTS,XP(I)+W(I)*XPS(ITYPE))
      GOTO 405
408   CONTINUE
415   JEP=JE+1
      DO 418 J=JEP,NPP
      I=NPP+JEP-J
      ITYPE=IABS(IPT(I))
      IF(XTL.LT.XP(I)+W(I)*XPS(ITYPE))GOTO 418
      JE=I
      XTL=DMAX1(XTL,XP(I)+W(I)*XPL(ITYPE))
      GOTO 415
```

```
418   CONTINUE

C *** CUTTING THE FITTED NUMBER OF PEAKS TO 10

420     IF(JE-JB+1.LT.10)GOTO 500
         IF(JE-JB+1.EQ.10.AND.IFTC.NE.0)GOTO 500
         ITY1=IABS(IPT(JE))
         ITY2=IABS(IPT(JB))
       IF(ITY1.LE.0.OR.ITY1.GE.5)PRINT*,' ARRAY PROB ITY1=',ITY1
       IF(ITY2.LE.0.OR.ITY2.GE.5)THEN
          PRINT*,' ARRAY PROB ITY2=',ITY2
          PRINT*,' JB,JE',JB,JE
       ENDIF
       IF(JE.LE.0.OR.JE.GE.256)PRINT*,' ARRAY PROB JE=',JE
       IF(JB.LE.0.OR.JB.GE.256)PRINT*,' ARRAY PROB JB=',JB
         IF((XT-XP(JE))/XPS(ITY1).LT.(XT-XP(JB))/XPL(ITY2))GOTO 450
       IF(XP(JE-1).LT.XT)GOTO 450
425      JE=JE-1
         GOTO 420

450    CONTINUE
       IF(XP(JB+1).GT.XT)GOTO 425
         JB=JB+1
         GOTO 420

C *** NOW THAT WE HAVE FOUND JB AND JE WE NEED TO RESTABLISH XTS AND XTL
C *** AND ADD THE OLD PEAKS TO THE STARTING GUESSES FOR THE NEW

500     CONTINUE
         IM=1
         IF(JB.GT.JE)GOTO 610
         IF(IFTC.NE.0)THEN
            XTS=1.D32
            XTL=-XTS
         ENDIF
        DO 600 I=JB,JE
        ECN(IM)=SC(I)
        EXN(IM)=SXP(I)
        EWN(IM)=SW(I)
        CN(IM)=C(I)
        XN(IM)=XP(I)
        WN(IM)=W(I)
      IF(IFTC.LT.0)WN(IM)=.5*WN(IM)
        IPTN(IM)=IPT(I)
         ITYPE=IABS(IPT(I))
        XTS=DMIN1(XTS,XN(IM)+XPS(ITYPE)*WN(IM))
        XTL=DMAX1(XTL,XN(IM)+XPL(ITYPE)*WN(IM))
        IM=IM+1
600     CONTINUE

C *** REMOVE THE NEW PEAKS FROM THE FILE LIST

610      IM=IM-1
       IF(IFTC.LT.0.AND.IM.EQ.0)RETURN
         IF(IM.EQ.0)GOTO 650
        NPP=NPP-IM
        DO 620 I=JB,NPP
        SC(I)=SC(I+IM)
```

```
        SXP(I)=SXP(I+IM)
        SW(I)=SW(I+IM)
        C(I)=C(I+IM)
        XP(I)=XP(I+IM)
        W(I)=W(I+IM)
620      IPT(I)=IPT(I+IM)

C *** UPDATING CHIS AND THE FILE FOR THE REMOVAL OF THESE TERMS

650     IB=XTS-.2*(XTL-XTS)-XOFF
        IE=XTL+.2*(XTL-XTS)-XOFF
        IB=MAX0(1,IB)
        IE=MIN0(N,IE)
      DO 700 I=IB,IE
700   CHIS=CHIS-(F(I)-FB(I))**2*WX(I)
       DO 750 I=IB,IE
      XI=I+XOFF
750   FB(I)=FB(I)-POLYA(XI,CN,XN,WN,IPTN,IM)

C *** GETTING READY TO CALL QMIN

         IF(IFTC.EQ.0.OR.(IFTC.EQ.1.AND.IM.EQ.0))THEN
            IM=IM+1
            CN(IM)=CNNT
            WN(IM)=WPNNT
            IPTN(IM)=INNT
            XN(IM)=XT
        ENDIF
        IQFL=0
        IBO=IB
        IEO=IE
5173    CONTINUE
        IF(IABS(IWF).NE.9)
     # CALL QMIN(FB,F,WX,IB,IE,IM,CN,XN,WN,IPTN,WFN,IWF,PCON,
     # FW,SFW,XOFF,IPW,ECN,EXN,EWN,XVB,XVE)
        IF(IWF.EQ.9)
     # CALL FQMIN(FB,F,WX,IB,IE,IM,CN,XN,WN,IPTN,WFN,IWF,PCON,
     # FW,SFW,XOFF,IPW,ECN,EXN,EWN,XVB,XVE)
        IF(IWF.EQ.-9)THEN
          CALL FNMIN(FB,F,WX,IB,IE,IM,CN,XN,WN,IPTN,WFN,IWF,PCON,
     #     FW,SFW,XOFF,IPW,ECN,EXN,EWN,XVB,XVE)
        ENDIF
C *** CHECKING TO SEE THAT WE ARE IN THE CORRECT RANGE
        IF(IQFL.EQ.1)GOTO 5184
        DO 5180 I=1,IM
        IPTT=XN(I)-XOFF-2*WN(I)
        IPTT=MAX0(1,IPTT)
        IF(IPTT.GE.IB)GOTO 5175
        IPC=XN(I)-XOFF+WN(I)
        IF(IPC.LT.IB)CN(I)=.1*SQRT(ABS(WX(IPTT)))
        IF(IPC.LT.IB)GOTO 5175
        IQFL=1
        IB=IPTT
5175    IPTT=XN(I)-XOFF+WN(I)
        IPTT=MIN0(IPTT,N)
        IF(IPTT.LE.IE)GOTO 5180
        IPC=XN(I)-XOFF-WN(I)
        IF(IPC.GT.IE)CN(I)=.1*SQRT(ABS(WX(IPTT)))
```

```
        IF(IPC.GT.IE)GOTO 5180
        IE=IPTT
        IQFL=1
5180    CONTINUE
        IF(IQFL.NE.1)GOTO 5184
        IF(IB.GE.IBO)GOTO 5182
      IBOM=IBO-1
        DO 5181 I=IB,IBOM
5181    CHIS=CHIS-(F(I)-FB(I))**2*WX(I)
5182    IF(IE.LE.IEO)GOTO 5173
      IEOP=IEO+1
        DO 5183 I=IEOP,IE
5183    CHIS=CHIS-(F(I)-FB(I))**2*WX(I)
        GOTO 5173
5184    CONTINUE
        IF(IM.EQ.0)GOTO 541
C FROM HERE TO 540 XN IS BEING INSERTED INTO XP IN SUCH A WAY AS TO KEEP
C IN ASCENDING ORDER
      DO 540 I=1,IM
       CN(I)=DABS(CN(I))
        JPT=XN(I)-XOFF
        WN(I)=DMAX1(1.25D0,WN(I))
        JPT=MAX0(1,MIN0(N,JPT))
        IF(JPT.EQ.N.OR.JPT.EQ.1)CN(I)=0.D0
        IF(IABS(IWF).EQ.9)GOTO 519
        IF(CN(I)*CN(I).LT.1./DSQRT(DABS(WX(JPT)*WN(I))))CN(I)=0.D0
        IF(CN(I).LT.1.D-12)GOTO 540
519   IF(NPP.LT.1)GOTO 532
520   CALL LOCATE(XN(I),XP,NPP,J)
      IF(J.EQ.NPP)GOTO 535
      JU=NPP+1
      JL=NPP
530   XP(JU)=XP(JL)
        WF(JU)=WF(JL)
      W(JU)=W(JL)
        IPT(JU)=IPT(JL)
      C(JU)=C(JL)
        SXP(JU)=SXP(JL)
        SW(JU)=SW(JL)
        SC(JU)=SC(JL)
      JU=JU-1
      JL=JL-1
      IF(JL-J)535,535,530
532   J=0
535   JP=J+1
        NPB=JP
        SC(JP)=ECN(I)
        SW(JP)=EWN(I)
        SXP(JP)=EXN(I)
      XP(JP)=XN(I)
        WF(JP)=WFN(I)
        IPT(JP)=IPTN(I)
      W(JP)=WN(I)
      C(JP)=CN(I)
      NPP=NPP+1
540   CONTINUE
541   CONTINUE
C CORRECTING THE FB AND RESIDUALS FOR THE NEW PEAK
```

```
      IF(NPP.EQ.0)RETURN
      DO 543 I=IB,IE
      XI=I+XOFF
        FB(I)=POLYA(XI,C,XP,W,IPT,NPP)
543   CHIS=CHIS+(F(I)-FB(I))**2*WX(I)
        RETURN
        END
C*******************************************************************************
      SUBROUTINE LOCATE(X,R,NMAX,J)
C*******************************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION R(1),IC(9)
      DATA IC/128,64,32,16,8,4,2,1,1/
      J=256
      DO 20 IL=1,9
      IF(J.GT.NMAX)GOTO 12
      IF(X.GT.R(J))GOTO 15
12    J=J-IC(IL)
      GOTO 20
15    J=J+IC(IL)
20    CONTINUE
      IF(J.EQ.0)RETURN
      IF(J.GT.NMAX)J=NMAX
      IF(X.LT.R(J))J=J-1
      RETURN
      END
C*******************************************************************************
        FUNCTION CEST(W,SW,ITT,F,FB,WX,XT,N,ERRSQ)
C*******************************************************************************
        IMPLICIT REAL*8 (A-H,O-Z)
        COMMON/SCONS/DU(150),XPS(5),XPL(5),NTYPE
        REAL*4 F(1),FB(1),WX(1)
C *** THIS CODE WAS DEVELOPED IN ANALOGY TO FITTING LOG(F**2) TO
C *** A GAUSSIAN, I.E. TO ALPHA + LAMBDA * (X-XC)**2
C *** WITH THE POLY USED INSTEAD OF THE GAUSSIAN (X-XC)**2 BECOMES
C *** ALOG(POLY)/LAMBDA**2, WHILE THE CONSTANT PARTIAL OF
C *** LAMBDA*(X-XC)**2 WITH RESPECT TO LAMBDA BECOMES (X-XC)*DPOLY/POLY
       WMIN=W-2*SW
       WMAX=W+2*SW
         NIT=0
         IBG=XT-WMAX
         IBG=MAX0(1,IBG)
         IEND=XT+WMAX
         IEND=MIN0(N,IEND)
         ERRSQT=1.D32
         DO 70 ICEN=1,5
5        P1=0
         P2=0
         A11=0
         A12=0
         A21=0
         A22=0
         XCEN=XT+.20*(ICEN-3)
         DO 10 I=IBG,IEND
         X=(I-XCEN)/W
         DF=(F(I)-FB(I))**2
         WT=WX(I)*DF
         ALF=0.D0
```

```
          IF(DF.GT.0.D0)ALF=DLOG(DF)
          P1=P1+ALF*WT
          A11=A11+WT
          P=DMAX1(1.D-4,POLYG(X,ITT,SP) )
          ALP=DLOG(P)
          A12=A12+ALP*WT
          SPSP=SP/P
          P2=P2+ALF*WT*SPSP*X
          A21=A21+SPSP*WT*X
          A22=A22+ALP*SPSP*WT*X
10        CONTINUE
        ALA=0
          A12=2*W*W*A12
          A22=2*W*W*A22
          IF(A22*A11.NE.A12*A21)ALA=(P2*A11-P1*A21)/(A22*A11-A12*A21)
          WEST=DMIN1(WMAX,1.01*W)
          IF(ALA.GT.0.D0)WEST=DMIN1(WMAX,DMAX1(WMIN,1.D0/DSQRT(ALA)))
          IF(DABS(WEST-W).GT..05.AND.NIT.LT.5)THEN
            W=WEST
            NIT=NIT+1
            GOTO 5
          ENDIF
          W=WEST
          ALP=DMIN1((P1-ALA*A12)/A11,50.D0)
          CEST=DEXP(.5*ALP)
          ERRSQ=0
          DO 50 I=IBG,IEND
          X=(I-XCEN)/W
          FA=CEST*POLYG(X,ITT,SP)
          ERRSQ=(F(I)-FB(I)-FA)**2*WX(I)+ERRSQ
50        CONTINUE
          ERRSQ=ERRSQ/(IEND-IBG+1)
          IF(ERRSQ.LT.ERRSQT)THEN
            ERRSQT=ERRSQ
            WTEMP=WEST
            XTEMP=XCEN
          ENDIF
          W=WTEMP
70        CONTINUE
          XT=XTEMP
          ERRSQ=ERRSQT
          RETURN
          END
C*************************************************************************
           FUNCTION POLYA(X,C,XP,W,IPT,NS)
C*************************************************************************
       IMPLICIT REAL*8(A-H,O-Z)
C THIS ROUTINE CALCULATES POLYA= SUM C(I)*STAN(I,X) USING POLYG FOR STAN
C NOTE THAT BY NOT HAVING XPS OR XPL LARGE, WE ASSUME THAT POLYB HAS BEE
C CALLED BEFORE THE FIRST CALL TO POLYA
      COMMON/SCONS/DU(150),XPS(5),XPL(5)
         SAVE /SCONS/
      DIMENSION C(1),XP(1),W(1),IPT(1)
      POLYA=0
      IF(NS.EQ.0)RETURN
      DO 200 J=1,NS
      XM=X-XP(J)
        ITEST=IABS(IPT(J))
```

```
      ISIGN=1
      IF(IPT(J).LE.0)ISIGN=-1
      IF(XM.LE.W(J)*XPS(ITEST).OR.XM.GE.W(J)*XPL(ITEST))GOTO 200
      XM=XM/W(J)
C  PARTIAL WRT C(J)
       FADD=ISIGN*C(J)*POLYG(XM,ITEST,SP)
      POLYA=POLYA+C(J)*FADD
200   CONTINUE
      RETURN
      END
C*********************************************************************
       FUNCTION POLYB(X,P,C,XP,W,IPT,NS)
C*********************************************************************
       IMPLICIT REAL*8(A-H,O-Z)
C THIS ROUTINE CALCULATES POLYB=SUM CI*SI ALONG WITH DPOLYB/DCI
C THE NEXT NS SI'S, WHICH HAVE DERIVATIVES WITH RESPECT TO C,XP,AND W
C ARE BSPLINES
         SAVE /SCONS/
      COMMON/SCONS/DU(150),XPS(5),XPL(5)
       DIMENSION P(1),C(1),XP(1),W(1),IPT(1)
      NC=1
5     NPART=1
      POLYB=0
      DO 200 J=1,NS
           ISIGN=1
      IF(IPT(J).LT.0)ISIGN=-1
        ITEST=IABS(IPT(J))
      P(NPART)=0.
      P(NPART+1)=0.
      P(NPART+2)=0.
      XM=X-XP(J)
      IF(XM.LE.W(J)*XPS(ITEST).OR.XM.GE.W(J)*XPL(ITEST))GOTO 200
      XM=XM/W(J)
C  PARTIAL WRT C(J)
       FADD=ISIGN*C(J)*POLYG(XM,ITEST,SP)
      P(NPART)=2.*FADD
      POLYB=POLYB+C(J)*FADD
C PARTIAL WRT XP(J)
      P(NPART+1)=-ISIGN*C(J)*C(J)*SP/W(J)
C PARTIAL WRT W(J)
      P(NPART+2)=P(NPART+1)*XM
200   NPART=NPART+3
      RETURN
      END
C*********************************************************************
      FUNCTION POLYG(X,IPT,SP)
C*********************************************************************
       IMPLICIT REAL*8(A-H,O-Z)
        CHARACTER*64 NA
      COMMON/SCONS/W(10,5),XP(10,5),C(10,5),XPS(5),XPL(5),IP,NC
C THIS ROUTINE CALCULATES POLYG=SUM CI*SI ALONG WITH DPOLY/DX
C IPT GIVES THE PEAK TYPE RANGING FROM 1 TO 5
        DIMENSION NST(5)
         SAVE
      IF(NC.EQ.1)GOTO 30
      NC=1
        IP=0
      OPEN(19,FILE='TEMP')
```

```
3       PRINT*,' ENTER THE NAME OF THE FILE WITH THE STANDARD PEAK'
        READ(2,'(A)')NA
      WRITE(19,'(A)')NA
      PRINT*,'ABOUT TO OPEN ',NA
        IF(NA.EQ.'\'.OR.NA.EQ.'STOP')GOTO 28
        OPEN(11,FILE=NA,STATUS='OLD',ERR=3)
      WRITE(*,102)
102   FORMAT(' THE FOLLOWING CONSTANTS,PEAKS,AND WIDTHS ARE THE',
     X ' STANDARD PEAK')
        IP=IP+1
        NS=1
        IF(IP.GT.5)GOTO 300
5     READ(11,101,END=20)C(NS,IP),XP(NS,IP),W(NS,IP)
101   FORMAT(5X,3E20.7)
      WRITE(*,103)NS,C(NS,IP),XP(NS,IP),W(NS,IP)
103   FORMAT(I5,3E20.7)
      NS=NS+1
      GOTO 5
20    NS=NS-1
        CLOSE(11)
      XPS(IP)=1.D32
      XPL(IP)=-1.D32
      DO 25 J=1,NS
      XPS(IP)=DMIN1(XPS(IP),XP(J,IP)-W(J,IP))
25    XPL(IP)=DMAX1(XPL(IP),XP(J,IP)+W(J,IP))
        NST(IP)=NS
        GOTO 3
28    CLOSE(19)
30    CONTINUE
       POLYG=0.
      SP=0.
        NS=NST(IPT)
      DO 200 J=1,NS
      XM=X-XP(J,IPT)
      IF(XM.LE.-W(J,IPT).OR.XM.GE.W(J,IPT))GOTO 200
      XM=XM/W(J,IPT)
      FADD=C(J,IPT)*((1.+XM)*(1.-XM))**3
      POLYG=POLYG+FADD
       SP=SP+3.*(FADD/(1.+XM)-FADD/(1.-XM))/W(J,IPT)
200   CONTINUE
      RETURN
300     PRINT*,' ATTEMPT TO DEFINE MORE THAN FIVE STANDARDS'
        STOP
      END
C*************************************************************************
        SUBROUTINE QMIN(FB,F,WX,IB,IE,NSC,C,XP,W,IPT,WF,IWF,PCON,FW,SFW
     # ,XOFF,IPP,EC,EX,EW,XVB,XVE)
C*************************************************************************
C FB ARE THE BACKGROUND VALUES, F THE MEASURED VALUES, FW THE
C FIT TO THE WIDTHS, C THE CONSTANTS FOR THE PEAKS, XP THE
C PEAK LOCATIONS, W THE PEAK WIDTHS.  THESE LAST THREE ARE
C RETURNED BY QMIN, BUT INITIAL ESTIMATES MUST BE SUPPLIED.
      IMPLICIT REAL*8(A-H,O-Z)
      REAL*4 FB,F,WX,FMIN,FMAX,FA,SQRT,AMAX1,AMIN1,ALOG10,XOFF
      DIMENSION IO(10),FB(1),F(1),WX(1),C(1),XP(1),W(1),WF(1),FW(3,5)
      DIMENSION IPT(1),P(30),CONS(30),SM(30),PC(30),PPCC(480),
     # AM(480),EC(1),EW(1),EX(1),SFW(5,5),AEWF(10),EFT(10),IPTS(10)
     # ,AEWPC(10),CS(10),ECS(10),WS(10),EWS(10),XPS(10),EXS(10)
```

```
        DATA PD/12./
        CHI1=1.E32
9       CHB=1.E31
        CHL=1.E32
        FR=0
        IEN=0
        NT=3*NSC
        NPART=1
         DO 24 J=1,NSC
        CONS(NPART)=C(J)
        SM(NPART)=1.D-2
        CONS(NPART+1)=XP(J)
        SM(NPART+1)=SM(NPART)*1.D6
        L=IABS(IPT(J))
        WPP=DSQRT(DMAX1(3D0,FW(1,L)+XP(J)*(FW(2,L)+
       # XP(J)*FW(3,L))))
        SW=DSQRT(DABS(SFW(1,L)+XP(J)*(SFW(2,L)+XP(J)*(SFW(3,L)
       # +XP(J)*(SFW(4,L)+XP(J)*SFW(5,L)))))
        SW=SW/(2*WPP)
        W(J)=DMAX1(1.5D0,WPP-2*SW,DMIN1(W(J),WPP+2*SW))
        CONS(NPART+2)=W(J)
        SM(NPART+2)=SM(NPART)*1.D4
24      NPART=NPART+3
          NDT=NT*(NT+1)/2
        NKIT=30
        KIT=1
28        CHI=0
           PEN=0
        DO 30 J=1,NT
30      PC(J)=0.
        DO 35 J=1,NDT
35      PPCC(J)=0.
        DO 200 I=IB,IE
        X=I+XOFF
        FA=POLYB(X,P,C,XP,W,IPT,NSC)+FB(I)
          RES=AMIN1(1.E12,AMAX1(-1.E12,F(I)-FA))
        CHI=CHI+RES**2*WX(I)
        W1=2*(FA-F(I))*WX(I)
        W2=2*WX(I)
          KJ=0
        DO 80 J=1,NT
        PC(J)=PC(J)+W1*P(J)
        W2P=W2*P(J)
        DO 80 K=1,J
          KJ=KJ+1
80      PPCC(KJ)=PPCC(KJ)+P(K)*W2P
200     CONTINUE

         CALL WADJ(FW,SFW,PC,PPCC,AM,
       #   PEN,W,XP,IPT,NSC,KIT)

           CHI=CHI+PEN
        IF(CHI.LE.1.01*CHL.AND.KIT.GE.NKIT)GOTO 2502
        IF(CHI.LE.1.01*CHL.AND.IEN.EQ.1)GOTO 2502
        IF(CHL-CHB.GT..1D0)GOTO 248
        IF(KIT.LE.3)GOTO 248
          IEN=1
        IF(CHI.LT.CHL)GOTO 2502
```

```
248    CALL SMSQ(CHI,CHB,CHL,PC,PPCC,AM,FR,CONS,SM,NT,IPP)
250    NPART=1
       DO 260 J=1,NSC
       C(J)=DMAX1(.1D0,CONS(NPART))
       XP(J)=DMAX1(1.D0*(IB+XOFF),DMIN1(1.D0*(IE+XOFF),CONS(NPART+1)))
       W(J)=DABS(CONS(NPART+2))
       L=IABS(IPT(J))
       WPP=DSQRT(DMAX1(3D0,FW(1,L)+XP(J)*(FW(2,L)+
      # XP(J)*FW(3,L))))
       SW=DSQRT(DABS(SFW(1,L)+XP(J)*(SFW(2,L)+XP(J)*(SFW(3,L)
      # +XP(J)*(SFW(4,L)+XP(J)*SFW(5,L)))))
       SW=SW/(2*WPP)
       W(J)=DMAX1(1.5D0,WPP-2*SW,DMIN1(W(J),WPP+2*SW))
260    NPART=NPART+3
       KIT=KIT+1
       IF(KIT.GT.2*NKIT)GOTO 280
       GOTO 28
2502   CONTINUE
         CHI=CHI-PEN
         DO 2503 I=1,NDT
2503     AM(I)=PPCC(I)
         CALL SMINV(AM,NT,IFL)
         DO 2510 I=1,NDT
2510     AM(I)=DMIN1(1.D9,DMAX1(-1.D9,AM(I)))
         CHIP=DMAX1(1.D0,CHI/MAX0(1,IE-IB+1-NT))
         DO 2520 J=1,NSC
         JC=1+3*(J-1)
         JX=JC+1
         JW=JX+1
         JCC=JC*(JC+1)/2
         JWW=JW*(JW+1)/2
         ICT=KIJ(JC,JC+2)
         ESTR2=(2*C(J)*W(J))**2*AM(JCC)
         ESTR2=ESTR2+4*C(J)**3*W(J)*AM(ICT)
         ESTR2=ESTR2+C(J)**4*AM(JWW)
         EC(J)=DSQRT(DMAX1(1.D-30,ESTR2*CHIP*2))
         EX(J)=DSQRT(DMAX1(1.D-30,2*AM(JX*(JX+1)/2)*CHIP))
         EX(J)=DMIN1(DMAX1(10.D0,2*W(J)),EX(J))
         EW(J)=DSQRT(DMAX1(1.D-30,2*AM(JW*(JW+1)/2)*CHIP))
         EW(J)=DMIN1(DMAX1(10.D0,2*W(J)),EW(J))
         WF(J)=W(J)
2520     CONTINUE
          IF(IPP.EQ.0)RETURN
         NP=IE-IB+1
       WRITE(*,104)NP
104    FORMAT(' WE ARE FITTING',I5,' POINTS')
       WRITE(*,102)CHI,PEN
102    FORMAT(' CHISQUARE IS',E20.7,' PEN IS',E20.7)
       IF(NSC.GT.0)WRITE(*,103)(I,C(I),EC(I),XP(I),EX(I),W(I),EW(
      # I),I=1,NSC)
103    FORMAT(' THE B SPLINE COEFFS'/'    #',8X,'C(I)',17X,'XP(I)',
      # 18X,'W(I)'/(I5,3(F12.4,F10.6)))
        RETURN
280    WRITE(*,105)CHI,CHB,CHL,(CONS(I),I=1,NT)
105    FORMAT(' LOOPING IN QMIN, CHI,CHB,CHL/CONS'/(3E20.12))
       NPART=1
       DO 290 J=1,NSC
       C(NPART)=0
```

```
290    NPART=NPART+3
       RETURN
        END
C*******************************************************************************
         SUBROUTINE FQMIN(FB,F,WX,IB,IE,NSC,C,XP,W,IPT,WF,IWF,PCON,FW,SFW
     # ,XOFF,IPP,EC,EX,EW,XVB,XVE)
C*******************************************************************************
C FB ARE THE BACKGROUND VALUES, F THE MEASURED VALUES, FW THE
C FIT TO THE WIDTHS, C THE CONSTANTS FOR THE PEAKS, XP THE
C PEAK LOCATIONS, W THE PEAK WIDTHS.  THESE LAST THREE ARE
C RETURNED BY QMIN, BUT INITIAL ESTIMATES MUST BE SUPPLIED.
       IMPLICIT REAL*8(A-H,O-Z)
       REAL*4 FB,F,WX,FMIN,FMAX,FA,SQRT,AMAX1,AMIN1,ALOG10,XOFF
       DIMENSION IO(10),FB(1),F(1),WX(1),C(1),XP(1),W(1),WF(1),FW(3,5)
       DIMENSION IPT(1),P(30),CONS(30),SM(30),PC(30),PPCC(480),
      # AM(480),EC(1),EW(1),EX(1),SFW(5,5),AEWF(10),EFT(10),IPTS(10)
      # ,AEWPC(10),CS(10),ECS(10),WS(10),EWS(10),XPS(10),EXS(10)
         DATA PD/12./
         CHI1=1.E32
9      CHB=1.E31
       CHL=1.E32
       FR=0
       IEN=0
       NT=NSC
       NPART=1
        DO 24 J=1,NSC
        C(J)=DMAX1(.1D0,C(J))
       CONS(NPART)=C(J)
       SM(NPART)=1.D-2
24     NPART=NPART+1
         NDT=NT*(NT+1)/2
       NKIT=30
       KIT=1
28       CHI=0
          PEN=0
       DO 30 J=1,NT
30     PC(J)=0.
       DO 35 J=1,NDT
35     PPCC(J)=0.
       DO 200 I=IB,IE
       X=I+XOFF
       FA=POLYB(X,P,C,XP,W,IPT,NSC)+FB(I)
         RES=AMIN1(1.E12,AMAX1(-1.E12,F(I)-FA))
       CHI=CHI+RES**2*WX(I)
       W1=2*(FA-F(I))*WX(I)
       W2=2*WX(I)
         KJ=0
       DO 80 J=1,NT
       JC=3*(J-1)+1
       PC(J)=PC(J)+W1*P(JC)
       W2P=W2*P(JC)
       DO 80 K=1,J
         KC=3*(K-1)+1
         KJ=KJ+1
80     PPCC(KJ)=PPCC(KJ)+P(KC)*W2P
200    CONTINUE
       IF(CHI.LE.1.01*CHL.AND.IT.GE.NKIT)GOTO 2502
       IF(CHI.LE.1.01*CHL.AND.LN.EQ.1)GOTO 2502
```

```
      IF(CHL-CHB.GT..1D0)GOTO 248
      IF(KIT.LE.3)GOTO 248
        IEN=1
      IF(CHI.LT.CHL)GOTO 2502
248   CALL SMSQ(CHI,CHB,CHL,PC,PPCC,AM,FR,CONS,SM,NT,IPP)
      DO 260 J=1,NSC
260   C(J)=DMAX1(.1D0,CONS(J))
      KIT=KIT+1
      IF(KIT.GT.2*NKIT)GOTO 280
      GOTO 28
2502  CONTINUE
        CHI=CHI-PEN
        DO 2503 I=1,NDT
2503    AM(I)=PPCC(I)
        CALL SMINV(AM,NT,IFL)
        DO 2510 I=1,NDT
2510    AM(I)=DMIN1(1.D9,DMAX1(-1.D9,AM(I)))
        CHIP=DMAX1(1.D0,CHI/MAX0(1,IE-IB+1-NT))
        DO 2520 J=1,NSC
        EC(J)=2*(2*C(J)*W(J))**2*AM(J*(J+1)/2)*CHIP
      EC(J)=DSQRT(DMAX1(1.D-30,EC(J)))
        EW(J)=0
        EX(J)=0
        WF(J)=W(J)
2520    CONTINUE
         IF(IPP.EQ.0)RETURN
        NP=IE-IB+1
      WRITE(*,104)NP
104   FORMAT(' WE ARE FITTING',I5,' POINTS')
      WRITE(*,102)CHI,PEN
102   FORMAT(' CHISQUARE IS',E20.7,' PEN IS',E20.7)
      IF(NSC.GT.0)WRITE(*,103)(I,C(I),EC(I),XP(I),EX(I),W(I),EW(
     # I),I=1,NSC)
103   FORMAT(' THE B SPLINE COEFFS'/'    #',8X,'C(I)',17X,'XP(I)',
     # 18X,'W(I)'/(I5,3(F12.4,F10.6)))
       RETURN
280   WRITE(*,105)CHI,CHB,CHL,(CONS(I),I=1,NT)
105   FORMAT(' LOOPING IN QMIN, CHI,CHB,CHL/CONS'/(3E20.12))
      NPART=1
      DO 290 J=1,NSC
290   C(J)=0
      RETURN
      END
C*******************************************************************************
        SUBROUTINE WADJ(FW,SFW,PC,PPCC,AI,
     #  PEN,W,XP,IPT,NSC,KIT)
C*******************************************************************************
        IMPLICIT REAL*8 (A-H,O-Z)
        DIMENSION PC(1),PPCC(1),FW(3,5),SFW(5,5),AI(1),
     # XP(1),IPT(1),W(1)
       NPART=3
          PEN=0.D0
          DO 210 J=1,NSC
             L=IABS(IPT(J))
       NPM22=(NPART-2)*(NPART-1)/2
       T11=DMAX1(1.D-12,DMIN1(1.D12,PPCC(NPM22)))
       NPM2NP=NPART-2+NPART*(NPART-1)/2
       T12=DMAX1(-1.D12,DMIN1(1.D12,PPCC(NPM2NP)))
```

```
      NPNP=NPART*(NPART+1)/2
      T22=DMAX1(1.D-12,DMIN1(1.D12,PPCC(NPNP)))
C     IF(XP(J).GT.XVB.AND.XP(J).LT.XVE)GOTO 210
      WPP=DSQRT(DMAX1(3D0,FW(1,L)+XP(J)*(FW(2,L)+
     # XP(J)*FW(3,L))))
      SW=DSQRT(DABS(SFW(1,L)+XP(J)*(SFW(2,L)+XP(J)*(SFW(3,L)
     # +XP(J)*(SFW(4,L)+XP(J)*SFW(5,L))))))
      SW=SW/(2*WPP)
      SW=DMAX1(.1D-2,SW)
      SW2=SW*SW
      DELTAW=(W(J)-WPP)
208   PT=DMAX1(0.D0,(2*T11/SW2+T12*T12-T11*T22)/(2*T11))
      PC(NPART)=2*PT*DELTAW+PC(NPART)
      PPCC(NPNP)=2*PT+PPCC(NPNP)
      PEN=PEN+PT*DELTAW*DELTAW
210   NPART=NPART+3
      RETURN
      END
      INCLUDE BREAD.FO
      INCLUDE ROPKS.FO
      INCLUDE CFEST.FOR
      INCLUDE BKGFIT.FO
      INCLUDE FWHM.FO
      INCLUDE RESL.FO
      INCLUDE SMSQ.FO
      INCLUDE CQMIN.FOR
      INCLUDE FNMIN.FOR
C*************************************************************************
C*************************************************************************
      SUBROUTINE BKGFIT(XOFF,F,FA,WX,N,CONS,NV,NVM,BKGF,LFLAG,NCALL,
     # CUT,FNAME)
C*************************************************************************

        IMPLICIT REAL*8 (A-H,O-Z)
       DIMENSION XC(512),FC(512),WC(512),FAC(512),
     # CONS(1),P(100),F(1),FA(1),WX(1)
      COMMON/CLFL/ILFLAG
        CHARACTER*64 FNAME
        CHARACTER*4 BKGF
         CHARACTER*1 ANS
        REAL*4 XOFF,F,FA,WX
      ILFLAG=LFLAG
      PRINT*,' IN BKGFIT, BKGF=',BKGF
        IF(BKGF.EQ.'NOBF'.AND.NCALL.GT.0)RETURN
      CALL FALPAB(ALP,A,B,CUT,ASHIFT,CHIL)
       PRINT*,' A,B,ALP,ASHIFT ',A,B,ALP,ASHIFT
         NITT=3
         IF(BKGF.EQ.'FITB')NITT=10
         NONEWK=0
         IF(BKGF.EQ.'NONK'.OR.NV.GE.NVM.OR.BKGF.EQ.'FINK')NONEWK=1
         IF(NCALL.GT.0)GOTO 60

C *** TRY TO READ THE OLD CONSTANTS AND SET FA TO WHERE IT WAS

      PRINT*,' IN BKGFIT FILE NAME IS ',FNAME
        IF(FNAME.EQ.'NONE')GOTO 40
         NONEWK=1
        OPEN(1,FILE=FNAME,STATUS='OLD',ERR=1197)
```

```
          READ(1,*)LFLAG
         ILFLAG=LFLAG
          NV=0
20        READ(1,*,END=25)CONS(NV+1)
          NV=NV+1
          PRINT*,NV,CONS(NV)
          GOTO 20
25        CLOSE(1)
           IF(CONS(NV-1).EQ.0.D0)GOTO 45
          DO 30 J=1,N
           XDP=XOFF+J
          CALL POLY(XDP,P,NV,CONS,FDP)
           FA(J)=FDP
30        CONTINUE
         NCALL=1
           IF(BKGF.NE.'NOBF')NCALL=30
           RETURN

40        CONTINUE
           NONEWK=1
           NV=4
         ILFLAG=LFLAG
C *** INITIALIZE FA
45         CONTINUE
          DO 50 I=1,N
          FA(I)=F(I)
50        CONTINUE
60        NN=((N-1)/16+1)
C *** ABOVE MAKES THE NUMBER OF POINTS A MULTIPLE OF 16
          WAVE=0
          XAVE=0
        FAVE=0
        DO 120 I=1,10
          WAVE=WAVE+WX(N-I+1)
          XAVE=XAVE+XOFF+N+1-I
120     FAVE=F(N-I+1)+FAVE
          WAVE=.1*WAVE
          XAVE=.1*XAVE
        FAVE=.1*FAVE
160       CONTINUE
          DO 500 ITT=1,NITT
C *** DATA COMPRESSION
          JC=1
          DO 200 I=1,NN
          FC(I)=0
          WC(I)=0
          XC(I)=0
          JCP=JC+15
          DO 180 J=JC,JCP
          XAD=XAVE*WAVE
          FAD=FAVE*WAVE
          WAD=WAVE
          WT=1
          Y=(F(J)-FA(J))**2*WX(J)
          IF(J.LT.N)THEN
          WT=(1+ALP*Y)*A
C *** NOTE THAT WX IS 1/ER**2 IN THE ORIGINAL DATA
          IF(F(J).GT.FA(J))WT=B/(1+ALP*Y)
```

```
C *** HELPING THE FIT AT THE BEGINNING
         IF(J.LT.10)WT=WT*4
        WAD=WX(J)*WT
        FAD=(F(J))*WAD
        XAD=(XOFF+J)*WAD
        ENDIF
C *** NOTE THAT WC WILL BE 1/ERR**2
        WC(I)=WC(I)+WAD
        XC(I)=XC(I)+XAD
180     FC(I)=FC(I)+FAD
        XC(I)=XC(I)/WC(I)
        FC(I)=FC(I)/WC(I)
         CALL POLY(XC(I),P,NV,CONS,FAC(I))
200     JC=JC+16

         IF(ITT.NE.1)GOTO 300
         IF(NONEWK.EQ.1)GOTO 300
        NV=NV+2
         CONS(NV-1)=0
         PRINT*,' BEFORE MAX3D NN,NV ',NN,NV
C        CALL MAX3D(ILR,XC,FC,FAC,WC,NN,NV)
        CALL BRESL(FC,FAC,WC,ILR,NN,LFLAG)
      ILR=MAX0(1,ILR)
      IUP=ILR+MAX0(5,NN/(2*NV))
        PRINT'('' NEW KNOTS AT'',G14.6)',XC(ILR),XC(IUP)
        CONS(NV)=XC(ILR)
      NV=NV+2
      CONS(NV-1)=0
      CONS(NV)=XC(IUP)
300     CONTINUE
        IF(BKGF.NE.'FIXK')THEN
        CALL SPFIT(XC,FC,FAC,WC,NN,NV,CONS)
      ELSE
        CALL SPFIXK(XC,FC,FAC,WC,NN,NV,CONS)
      ENDIF
      PRINT*,' ITERATION NUMBER',ITT,' OF ',NITT
        WRITE(*,1190)NN,NN*CHIL,(CONS(I),I=1,NV)
1190    FORMAT(' BKG ,CHIL',I3,F5.0/(6E14.6))
C *** REMOVING UNUSED CONSTANTS
      DO 350 I=6,NV,2
      IF(CONS(I)-XOFF.GT.0.D0)GOTO 350
      NV=NV-2
      IM=I-1
      DO 340 J=IM,NV
340   CONS(J)=CONS(J+2)
350   CONTINUE
C *** DATA EXPANSION
        DO 430 J=1,N
         XDP=XOFF+J
        CALL POLY(XDP,P,NV,CONS,FDP)
         FA(J)=FDP
430     CONTINUE
500     CONTINUE
600     CONTINUE
        NCALL=1
        RETURN
1197  PRINT*,' COULD NOT OPEN FILE= ',FNAME
      READ(*,*)ITEST
```

```
      STOP
      END
C*************************************************************************
        SUBROUTINE POLY(X,P,NV,CONS,FA)
C*************************************************************************
C *** THE FIRST FOUR COEFFICIENTS REPRESENT A CUBIC
C *** THE REST ARE IN THE FORM C(I)*(C(I+1)-X)+ **3
        IMPLICIT REAL*8 (A-H,O-Z)
        DIMENSION P(NV),CONS(NV)
      COMMON/CLFL/ILFLAG
        NVS=NV-4
        P(1)=1.D0
        FA=CONS(1)
        DO 5 I=2,4
        P(I)=X*P(I-1)
5       FA=FA+CONS(I)*P(I)
        DO 20 I=5,NV,2
        P(I)=0.D0
        P(I+1)=0.D0
        IF(CONS(I+1).LE.X)GOTO 20
        DIFF=CONS(I+1)-X
        DIFF2=DIFF*DIFF
        P(I+1)=3*CONS(I)*DIFF2
        P(I)=DIFF2*DIFF
        FA=FA+CONS(I)*P(I)
20      CONTINUE
      IF(ILFLAG.EQ.0)RETURN
      FA=DEXP(DMIN1(FA,60.D0))
      DO 30 I=1,NV
30    P(I)=FA*P(I)
        RETURN
        END
C*************************************************************************
        SUBROUTINE SPFIT(XC,FC,FAC,WC,N,NV,CONS)
C*************************************************************************
        IMPLICIT REAL *8 (A-H,O-Z)
        CHARACTER*4 CHAR
C *** THIS ROUTINE MINIMIZES A SPLINE FIT TO THE COMPRESSED
C *** BACKGROUND
        DIMENSION PPCC(5050),PPCCI(5050),P( 100),PC( 100),CONS(NV)
        DIMENSION SM( 100)
        DIMENSION XC(1  ),FC(1  ),WC(1  ),FAC(1  )
        DATA NE/0/
      ALAM=25.6D0*N**3/NV**3
      IF(NV.GT.4)PRINT*,' ALAM=',ALAM
        IF(NE.GT.0)GOTO 14
        DO 10 I=1,4
10      SM(I)=1.D1
        DO 12 I=5,100,2
        SM(I)=1.D1
12      SM(I+1)=1.D4
14      FR=0.5
        CHB=1.E32
        CHL=1.E33
        NVD=NV*(NV+1)/2
         NIT=1
1045    CONTINUE
        DO 13 I=1,NV
```

```
13      PC(I)=0.
        DO 15 I=1,NVD
15       PPCC(I)=0.
         DO 18 I=6,NV,2
18       CONS(I)=DABS(CONS(I))
22       CHI=0
        DO 38 IT=1,N
        CALL POLY(XC(IT),P,NV,CONS,FAC(IT))
        ERR=(FC(IT)-FAC(IT))
        ERRS=ERR*ERR*WC(IT)
        CHI=CHI+ERRS
        K=0
        DO 25 I=1,NV
25     PC(I)=PC(I)-2*ERR*P(I)*WC(IT)
        W3=2*WC(IT)
        DO 30 I=1,NV
        W3T=W3*P(I)
        DO 30 J=1,I
        K=K+1
30     PPCC(K)=PPCC(K)+W3T*P(J)
38      CONTINUE
       NKK=(NV-4)/2
       DO 42 I=1,NKK
       NCOEF=4+I*2
       ISDD=KIJ(NCOEF,NCOEF)
       DO 40 J=1,NKK
       IF(I.EQ.J)GOTO 40
       NCNN=4+J*2
       DIFF=(CONS(NCOEF)-CONS(NCNN))
       DIFF2=DIFF*DIFF
       CHI=CHI+.5*ALAM/DIFF2
       DIFF3=DIFF2*DIFF
       PC(NCOEF)=PC(NCOEF)-2*ALAM/DIFF3
       ISD=KIJ(NCOEF,NCNN)
       DIFF4=DIFF*DIFF3
       PPCC(ISD)=PPCC(ISD)-6*ALAM/DIFF4
       PPCC(ISDD)=PPCC(ISDD)+6*ALAM/DIFF4
40     CONTINUE
42     CONTINUE
          PRINT'(A,3G20.6)',' SPFIT CHI,CHB,CHL',CHI,CHB,CHL
          G44TE=CHI-CHL-.01
         IF(G44TE.GT.0.D0 )GOTO 44
          IF(NIT.GT.200)GOTO 50
          NIT=NIT+1
          G50TE1=CHB-CHL+.01
          IF(G50TE1.GT.0.D0)GOTO 50
          G50TE2=G50TE1/CHB
          G50TE2=DABS(G50TE2)
          IF(G50TE2.LT.1.D-6)GOTO 50
          IF(DABS(CHL-CHB).LT.1.D-2.OR.DABS((CHL-CHB)/CHB).LT.1.D-6)
      # GOTO 50
44      CONTINUE
         CALL SMSQ(CHI,CHB,CHL,PC,PPCC,PPCCI,FR,CONS,SM,NV,0)
45       CONTINUE
          GOTO 1045
50      CONTINUE
         CHIT=CHI
         NE=NE+1
```

```
         PRINT'(A,G11.4)',' AT END OF SPFIT CHIT=',CHIT
         RETURN
       END
C*************************************************************************
       SUBROUTINE SPFIXK(XC,FC,FAC,WC,N,NV,CONS)
C*************************************************************************
       IMPLICIT REAL *8 (A-H,O-Z)
       CHARACTER*4 CHAR
C *** THIS ROUTINE MINIMIZES A SPLINE FIT TO THE COMPRESSED
C *** BACKGROUND
       DIMENSION PPCC(5050),PPCCI(5050),P( 100),PC( 100),CONS(NV)
       DIMENSION SM( 100),TCONS(100)
       DIMENSION XC(1  ),FC(1  ),WC(1  ),FAC(1  )
       DATA NE/0/
      ALAM=25.6D0*N**3/NV**3
      IF(NV.GT.4)PRINT*,' ALAM=',ALAM
      NVT=(NV-4)/2+4
       DO 10 I=1,NVT
10     SM(I)=1.D1
       FR=0.5
       CHB=1.E32
       CHL=1.E33
       NVD=NVT*(NVT+1)/2
C      DO 45 NIT=1,10
        NIT=1
1045   CONTINUE
       DO 13 I=1,NVT
13     PC(I)=0.
       DO 15 I=1,NVD
15      PPCC(I)=0.
      DO 18 I=1,4
18    TCONS(I)=CONS(I)
      DO 20 I=5,NVT
20    TCONS(I)=CONS(3+2*(I-4))
         CHI=0
       DO 38 IT=1,N
       CALL POLY(XC(IT),P,NV,CONS,FAC(IT))
       ERR=(FC(IT)-FAC(IT))
       ERRS=ERR*ERR*WC(IT)
       CHI=CHI+ERRS
       K=0
       DO 25 I=1,NVT
      IARG=I
      IF(I.GT.5)IARG=3+2*(I-4)
25    PC(I)=PC(I)-2*ERR*P(IARG)*WC(IT)
       W3=2*WC(IT)
       DO 30 I=1,NVT
      IARG=I
      IF(I.GT.5)IARG=3+2*(I-4)
       W3T=W3*P(IARG)
       DO 30 J=1,I
      JARG=J
      IF(J.GT.5)JARG=3+2*(J-4)
       K=K+1
30    PPCC(K)=PPCC(K)+W3T*P(JARG)
38     CONTINUE
        PRINT'(A,3G20.6)',' SPFIXK CHI,CHB,CHL',CHI,CHB,CHL
        G44TE=CHI-CHL-.01
```

```
        IF(G44TE.GT.0.D0 )GOTO 44
         IF(NIT.GT.100)GOTO 50
         NIT=NIT+1
         G50TE1=CHB-CHL+.01
         IF(G50TE1.GT.0.D0)GOTO 50
         G50TE2=G50TE1/CHB
         G50TE2=DABS(G50TE2)
         IF(G50TE2.LT.1.D-6)GOTO 50
         IF(DABS(CHL-CHB).LT.1.D-2.OR.DABS((CHL-CHB)/CHB).LT.1.D-6)
     # GOTO 50
44       CALL SMSQ(CHI,CHB,CHL,PC,PPCC,PPCCI,FR,TCONS,SM,NVT,0)
45       CONTINUE
      DO 48 I=1,NVT
      IARG=I
      IF(I.GT.5)IARG=3+2*(I-4)
48    CONS(IARG)=TCONS(I)
         GOTO 1045
50    CONTINUE
        CHIT=CHI
        NE=NE+1
         PRINT'(A,G11.4)',' AT END OF SPFIT CHIT=',CHIT
         RETURN
      END
C*************************************************************************
      SUBROUTINE FALPAB(ALP,A,B,CUT,ASHIFT,CHIL)
C*************************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION XG1(5),WG1(5),XG2(5),WG2(5),XG0(5),WG0(5)
      DATA XG1,WG1/.5133812615D0,.1188866291D1,.189642447D1,
     # .2661918482D1,.355390392D1,.260877802D0,.1993334055D0,
     # .3797122484D-1,.180587934D-2,.1168498619D-4/
      DATA XG2,WG2/.6568095668D0,.1326557084D1,.2025948015D1,
     # .2783290099D1,.3668470847D1,.185225285D0,.2062921868D0,
     # .4888991002D-1,.2686707670D-2,.1937314074D-4/
        DATA XG0,WG0/.3429013272D0,1.036610829D0,1.756683649D0,
     # 2.532731674D0,3.436159118D0,.6108626337D0,.2401386110D0,
     # .3387439445D-1,.1343645746D-2,.7640432855D-5/
        TSPI=.636619772
        C15=15.D0*DSQRT(TSPI)
        NC=0
        OALP=0
        ALP=.1D-4
        OFTZER=-2*CUT
C *** A AND B KEEP THE AVERAGE CORRECT FOR A GAUSSIAN DISTRIBUTION NO
C *** MATTER WHAT THE VALUE OF ALP.  AN OVERALL INCREASE OF BOTH A AND
C *** B ALSO KEEPS CHI**2 CORRECT
10      CONTINUE
      NC=NC+1
C *** B AND C ARE DETERMINED BY GAUSSIAN QUADRATURE
        BI=0
        C=0
        DI=0
        DO 20 I=1,5
        X2=XG1(I)*XG1(I)
        T1=1/(1+2*ALP*X2)
        BI=BI+WG1(I)*T1*(1-2*ALP*X2*T1)
        DI=DI+WG0(I)/(1+2*ALP*XG0(I)*XG0(I))
        DI=DI-20*WG2(I)/(1+2*ALP*XG2(I)*XG2(I))**2
```

```
      DI=DI+32*WG2(I)*XG2(I)*XG2(I)/(1+2*ALP*XG2(I)*XG2(I))**3
20    C=C+WG2(I)/(1+2*ALP*XG2(I)*XG2(I))
      BI=4*BI
C *** 1.128379 IS 2/PI**.5
      C=C*1.1283792D0
      DI=DI*15*1.1283792D0
C *** 2=A(1+3ALP)+2*B*C
      A=1
      B=(2-A*(1+3*ALP))/(2*C)
      C2=B*DI+30*A*(1+6*ALP)
      TCUT=1/(1+ALP*CUT*CUT)
      C1=2*B*CUT*TCUT*(1-2*CUT*ALP*TCUT)
      ASHIFT=-C1/C2
      CHIL=.5*C1*C1/C2
      FTZER=C15*(A*(2+8*ALP)-B*BI)-C1
      FP=(FTZER-OFTZER)/(ALP-OALP)
      OALP=ALP
      ALP=ALP-FTZER/FP
      OFTZER=FTZER
      IF(NC.GT.250)THEN
        PRINT*,' CANNOT FIND ALP IN FALPAB'
        STOP
      ENDIF
      IF(DABS((ALP-OALP)/ALP).GT.1.D-7)GOTO 10
      RETURN
      END
C*********************************************************************
       SUBROUTINE BRESL(F,FB,WX,ILR,N,LFLAG)
C*********************************************************************
       IMPLICIT REAL*8 (A-H,O-Z)
       DIMENSION F(1),WX(1),FB(1)
20    ALR=0
        NM2=N-2
        FSP2=(F(2)-FB(2))*DSQRT(WX(2))
          FSP1=(F(1)-FB(1))*DSQRT(WX(1))
      FS=0
        FSM1=0
        FSM2=0
        FSM3=0
        SUM=FSP1+FSP2
      DO 40 I=1,N
        FSM3=FSM2
        FSM2=FSM1
        FSM1=FS
        FS=FSP1
        FSP1=FSP2
        FSP2=0
        IF(I.LT.NM2)THEN
        FSP2=(F(I+2)-FB(I+2))*DSQRT(WX(I+2))
      ENDIF
        SUM=SUM+FSP2-FSM3
      IF(ALR.GT.DABS(SUM))GOTO 40
      ILR=I
      ALR=DABS(SUM)
40    CONTINUE
      RETURN
      END
C*********************************************************************
```

```
        SUBROUTINE BREAD(XE,F,IIDAT,WORDBYTE,WX,NP,EXTWT,NA,NAWT,N1,N2)
C*************************************************************************
        DIMENSION F(1),WX(1),IIDAT(1)
        CHARACTER*4 II(40),EXTWT
      CHARACTER*1 ANS
        CHARACTER*64 NA,NAWT
      INTEGER*1 WORDBYTE(1),B1,B2,B3,B4
C *** F(N) IS THE SPECTRUM CURRENTLY BEING FITTED.
        PRINT*,' IN BREAD NA IS ',NA
        II(1)=NA(:4)
        II(2)=NA(5:8)
1943    FORMAT(A)
        WRITE(*,1178)N1,N2
        N2=MIN0(N2,8191+N1)
1178    FORMAT(' IN BREAD N1,N2',2I5)
        IT=INDEX(NA,'.')
        IRF=0
      PRINT*,' IN BREAD IT=',IT
      PRINT*,' NA IS ',NA
         IF(NA(IT+1 :IT+3).EQ.'UF')GOTO 1100
       PRINT*,' TESTING FOR SALLY ',NA(IT+1:IT+5)
       IF(NA(IT+1 :IT+5).EQ.'SALLY')GOTO 2000
        IF(NA(IT+1:).EQ.'LEO'.OR.NA(IT+1:).EQ.'leo')IRF=1
        WRITE(*,*)IRF
        IF(IRF.EQ.0)OPEN(8,FILE=NA,STATUS='OLD',ERR=1320)
        IF(IRF.EQ.1)OPEN(8,FILE=NA,STATUS='OLD',
     # FORM='UNFORMATTED',ERR=1320)
        IF(IRF.NE.1)GOTO 420
        NREC=(N2-1)/128+1
        NU=128*NREC
        DO 416 III=1,NU,128
416     READ(8)(IIDAT(I),I=III,III+127)
        N=N2-N1+1
        XE=N2-N
C *** PERMUTING THE WORDS FROM VAX TO MAC STORAGE
      N4=4*N2
      DO 418 I = 1,N4,4
          B1= WORDBYTE(I)
          B2 = WORDBYTE(I+1)
          B3 = WORDBYTE(I+2)
          B4 = WORDBYTE(I+3)
          WORDBYTE(I)   = B4
          WORDBYTE(I+1) = B3
          WORDBYTE(I+2) = B2
          WORDBYTE(I+3) = B1
418   CONTINUE
C *** END OF WORD PERMUTING
        DO 400 I=1,N
        F(I)=IIDAT(I+N1-1)
        WX(I)=1./AMAX1(1.,F(I))
400     CONTINUE
        NP=N
        CLOSE(8)
        IF(EXTWT.EQ.'NORM')RETURN
C ***  I AM GOING TO ASSUME THAT THE WEIGHT FILE IS ALSO LEO
        OPEN(8,FILE=NAWT,STATUS='OLD',
     # FORM='UNFORMATTED',ERR=1340)
        NREC=(N2-1)/128+1
```

```
        NU=128*NREC
        DO 4161 III=1,NU,128
4161     READ(8)(IIDAT(I),I=III,III+127)
        N=N2-N1+1
        XE=N2-N
C *** PERMUTING THE WORDS FROM VAX TO MAC STORAGE
      N4=4*N2
      DO 4181 I = 1,N4,4
          B1= WORDBYTE(I)
          B2 = WORDBYTE(I+1)
          B3 = WORDBYTE(I+2)
          B4 = WORDBYTE(I+3)
          WORDBYTE(I)   = B4
          WORDBYTE(I+1) = B3
          WORDBYTE(I+2) = B2
          WORDBYTE(I+3) = B1
4181  CONTINUE
C *** END OF WORD PERMUTING
        DO 4001 I=1,N
        FT=IIDAT(I+N1-1)
        WX(I)=1./AMAX1(1.,FT)
4001     CONTINUE
      RETURN
420     READ(8,101)II
         WRITE(*,'(A,1X,A)')' II(40)=',II(40)
101     FORMAT(10A4)
      IF(II(40).EQ.'FREE')GOTO 2200
        IF(II(40).NE.'HDAT'.AND.II(40).NE.'IDAT'.AND.II(40).NE.'Z4DA'.
     # AND.II(40).NE.'I88D')
     # GOTO 195
        IF(II(40).NE.'Z4DA')READ(8,*)NXB,NEND
        NMULT=1
        IF(II(40).EQ.'Z4DA')READ(8,*)NXB,NEND,NMULT
      IF(II(40).EQ.'I88D')THEN
         NXB=N1-NXB+1
         IF(NXB.LT.1)THEN
            PRINT*,' YOU HAVE REQUESTED A FIT BEGINNING WITH',N1
            PRINT*,' WHICH IS LOWER THAN THE LOWEST CHANNEL STORED'
            PRINT*,' IN THE FILE ',NA
            READ(*,*)ITEST
            STOP
          ENDIF
      ELSE
        NXB=N1
      ENDIF
        NEND=N2
        I10=10
        IF(II(40).EQ.'Z4DA')I10=20
      IF(II(40).EQ.'I88D')I10=8
        NXBS=(NXB-1)/I10
        IF(NXBS.EQ.0)GOTO 145
        DO 144 I=1,NXBS
144     READ(8,156)
145     NIR=NEND-I10*NXBS
        NIR=MIN0(NIR,8192)
        IF(II(40).EQ.'HDAT')READ(8,156)(IIDAT(I),I=1,NIR)
        IF(II(40).EQ.'IDAT')READ(8,157)(IIDAT(I),I=1,NIR)
        IF(II(40).EQ.'Z4DA')READ(8,158)(IIDAT(I),I=1,NIR)
```

```
      IF(II(40).EQ.'I88D')READ(8,159)(IIDAT(I),I=1,NIR)
156     FORMAT(Z7,   9Z8)
157     FORMAT(10I8)
158     FORMAT(20Z4)
159   FORMAT(8I8)
        CLOSE(8)
        NP=NEND-NXB+1
        NADD=NXB-1-I10*NXBS
        XE=NEND-NP
        DO 190 I=1,NP
        F(I)=IIDAT(I+NADD)*NMULT
190     WX(I)=1./AMAX1(1.,F(I))
        RETURN
195      N=0
        NS=N1-1
        DO 197 I=1,NS
        READ(8,*)
197     CONTINUE
        N2=N2-N1+1
200   N=N+1
      IF(N.GT.N2)GOTO 220
      READ(8,*,END=220)XT,F(N),ER
         IF(N.EQ.1)XE=XT
        IF(ER.EQ.0.D0)WX(N)=1./AMAX1(1.,F(N))
        IF(ER.NE.0.D0)WX(N)=1./(ER*ER)
        GOTO 200
220   NP=N-1
        CLOSE(8)
        RETURN
1100     CONTINUE
      PRINT*,' BEFORE OPEN'
         OPEN(12,FILE=NA,STATUS='OLD',FORM='UNFORMATTED',ERR=1320)
       PRINT*,' AFTER OPEN'
         READ(12)N,XE
       PRINT*,' N,XE',N,XE
       PRINT*,' BEFORE READ'
         READ(12)(F(I),I=1,N)
         PRINT*,' IN BREAD N,XE',N,XE,'TI = ',F(30)
         N1=N1-XE
         NT=N+XE
         NC=MIN0(NT,N2)
         N2=N2-XE
         N1=MAX0(1,N1)
        N=N2-N1+1
        XE=N2-N
        DO 1120 I=1,N
        F(I)=F(I+N1-1)
        WX(I)=1./AMAX1(1.,F(I))
1120    CONTINUE
         NP=N
         CLOSE(12)
       IF(EXTWT.EQ.'NORM')RETURN
1140   CONTINUE
         OPEN(8,FILE=NAWT,STATUS='OLD',FORM='UNFORMATTED')
         READ(8)NDUM,DUM
         READ(8)(WX(I),I=1,NDUM)
       DO 1220 I=1,N
       WX(I)=1./AMAX1(1.,WX(I+N1-1))
```

```
1220  CONTINUE
      RETURN
1320  PRINT*,' COULD NOT OPEN FILE ',NA
      READ(*,*)ITEST
      STOP
1340  PRINT*,' COULD NOT OPEN FILE ',NAWT
      READ(*,*)ITEST
      STOP
2000  CONTINUE
         OPEN(8,FILE=NA,STATUS='OLD',FORM='UNFORMATTED')
       PRINT*,' IN SALLY PART OF BREAD'
       READ(8)(IIDAT(I),I=1,128)
       I1=0
      DO 2010 I=1,300
      READ(8,END=2015)(IIDAT(I1+J),J=1,128)
      I1=I1+128
2010  CONTINUE
2015  N=I1+J-1
      PRINT*,' AFTER READING DATA N IS',N
        NXB=N1
        NEND=N2
        NP=NEND-NXB+1
        NADD=NXB-1
        XE=NEND-NP
        DO 2190 I=1,NP
        F(I)=IIDAT(I+NADD)
2190    WX(I)=1./AMAX1(1.,F(I))
      RETURN
2200    CONTINUE
      DO 2210 I=1,15
        READ(8,'(10A4)')II
2210  WRITE(*,'(1X,10A4)')II

        NMULT=1
        NXB=N1
        NEND=N2
      I10=8
        NXBS=(NXB-1)/I10
        IF(NXBS.EQ.0)GOTO 2245
        DO 2244 I=1,NXBS
2244     READ(8,156)
2245     NIR=NEND-I10*NXBS
        NIR=MIN0(NIR,8192)
        READ(8,*)(IIDAT(I),I=1,NIR)
        CLOSE(8)
      PRINT*,'FIRST VALUE',IIDAT(1)
        NP=NEND-NXB+1
        NADD=NXB-1-I10*NXBS
        XE=NEND-NP
        DO 2290 I=1,NP
        F(I)=IIDAT(I+NADD)*NMULT
2290     WX(I)=1./AMAX1(1.,F(I))

      RETURN
      END
C*******************************************************************************
      FUNCTION CQMIN(FB,F,WX,XOFF,N,XPG,WG,L,FW,SFW,ERRSQ)
C*******************************************************************************
```

```
C FB ARE THE BACKGROUND VALUES, F THE MEASURED VALUES, FW THE
C FIT TO THE WIDTHS,  XP THE PEAK LOCATIONS, W THE PEAK WIDTHS.
C THESE LAST THREE ARE RETURNED, BUT INITIAL ESTIMATES MUST BE SUPPLIED.
      IMPLICIT REAL*8(A-H,O-Z)
      REAL*4 FB,F,WX,SQRT,ABS,XOFF
      DIMENSION FB(1),F(1),WX(1),FW(3,5),IPT(2),SFW(5,5)
      DIMENSION P(3),CONS(3),SM(3),PC(3),PPCC(6),AM(6),C(2),W(2),XP(2)
9     CHB=1.E31
      CHL=1.E32
      IPT(1)=L
      FR=0
      IEN=0
      NT=3
      IX=XPG-XOFF
      C(1)=SQRT(ABS(F(IX)-FB(IX)))
      CONS(1)=C(1)
      SM(1)=1.D-2
      XP(1)=XPG
      CONS(2)=XPG
      SM(2)=SM(1)*1.D6
      WPP=DSQRT(DMAX1(2D0,FW(1,L)+XPG*(FW(2,L)+
     # XPG*FW(3,L))))
      W(1)=WPP
      SW=DSQRT(DABS(SFW(1,L)+XPG*(SFW(2,L)+XPG*(SFW(3,L)
     # +XPG*(SFW(4,L)+XPG*SFW(5,L))))))
      SW=SW/(2*WPP)
      CONS(3)=W(1)
      SM(3)=SM(2)*1.D4
        NDT=NT*(NT+1)/2
      IB=XPG-2*W(1)-2*SW-XOFF
      IB=MAX0(1,IB)
      IE=XPG+2*W(1)+2*SW-XOFF
      IE=MIN0(N,IE)
      NKIT=30
      KIT=1
28      CHI=0
         PEN=0
      DO 30 J=1,NT
30    PC(J)=0.
      DO 35 J=1,NDT
35    PPCC(J)=0.
      DO 200 I=IB,IE
      X=I+XOFF
      FA=POLYB(X,P,C,XP,W,IPT,1)+FB(I)
        RES=AMIN1(1.E12,AMAX1(-1.E12,F(I)-FA))
      CHI=CHI+RES**2*WX(I)
      W1=2*(FA-F(I))*WX(I)
      W2=2*WX(I)
        KJ=0
      DO 80 J=1,NT
      PC(J)=PC(J)+W1*P(J)
      W2P=W2*P(J)
      DO 80 K=1,J
        KJ=KJ+1
80    PPCC(KJ)=PPCC(KJ)+P(K)*W2P
200   CONTINUE

       CALL WADJ(FW,SFW,PC,PPCC,AM,
```

```
     #  PEN,W,XP,IPT,1,KIT)

         CHI=CHI+PEN
       IF(CHI.LE.1.01*CHL.AND.KIT.GE.NKIT)GOTO 2502
       IF(CHI.LE.1.01*CHL.AND.IEN.EQ.1)GOTO 2502
       IF(CHL-CHB.GT..1D0)GOTO 248
       IF(KIT.LE.3)GOTO 248
         IEN=1
       IF(CHI.LT.CHL)GOTO 2502
248    CALL SMSQ(CHI,CHB,CHL,PC,PPCC,AM,FR,CONS,SM,NT,0)
250    NPART=1
       C(1)=DMAX1(.1D0,CONS(NPART))
       XP(1)=DMAX1(1.D0*(IB+XOFF),DMIN1(1.D0*(IE+XOFF),CONS(2)))
       W(1)=DMAX1(1.5D0,WPP-2*SW,DMIN1(CONS(3),WPP+2*SW))
       KIT=KIT+1
       IF(KIT.GT.2*NKIT)GOTO 280
       GOTO 28
2502   CONTINUE
         CHI=CHI-PEN
       ERRSQ=CHI/(IE+1-IB)
       CQMIN=C(1)
       WG=W(1)
       XPG=XP(1)
        RETURN
280    WRITE(*,105)CHI,CHB,CHL,(CONS(I),I=1,NT)
105    FORMAT(' LOOPING IN CQMIN, CHI,CHB,CHL/CONS'/(3E20.12))
       NPART=1
       CQMIN=0
       ERRSQ=1.D32
       RETURN
        END
C*******************************************************************************
         SUBROUTINE FNMIN(FB,F,WX,IB,IE,NSC,C,XP,W,IPT,WF,IWF,PCON,FW,SFW
     # ,XOFF,IPP,EC,EX,EW,XVB,XVE)
C*******************************************************************************
C FB ARE THE BACKGROUND VALUES, F THE MEASURED VALUES, FW THE
C FIT TO THE WIDTHS, C THE CONSTANTS FOR THE PEAKS, XP THE
C PEAK LOCATIONS, W THE PEAK WIDTHS.  THESE LAST THREE ARE
C RETURNED BY QMIN, BUT INITIAL ESTIMATES MUST BE SUPPLIED.
       IMPLICIT REAL*8(A-H,O-Z)
       REAL*4 FB,F,WX,FMIN,FMAX,FA,SQRT,AMAX1,AMIN1,ALOG10,XOFF
       DIMENSION IO(10),FB(1),F(1),WX(1),C(1),XP(1),W(1),WF(1),FW(3,5)
       DIMENSION IPT(1),P(30),CONS(30),SM(30),PC(30),PPCC(480),
     # AM(480),EC(1),EW(1),EX(1),SFW(5,5),AEWF(10),EFT(10),IPTS(10)
     # ,AEWPC(10),CS(10),ECS(10),WS(10),EWS(10),XPS(10),EXS(10)
         DATA PD/12./
         CHI1=1.E32
9      CHB=1.E31
       CHL=1.E32
       FR=0
       IEN=0
       NT=NSC
       NPART=1
        DO 24 J=1,NSC
       C(J)=DMAX1(.1D0,C(J))
       IPT(J)=1
       CONS(NPART)=C(J)
       SM(NPART)=1.D-2
```

```
24     NPART=NPART+1
         NDT=NT*(NT+1)/2
       NKIT=30
       KIT=1
       KSS=0
28       CHI=0
          PEN=0
       DO 30 J=1,NT
30     PC(J)=0.
       DO 35 J=1,NDT
35     PPCC(J)=0.
       DO 200 I=IB,IE
       X=I+XOFF
       FA=POLYB(X,P,C,XP,W,IPT,NSC)+FB(I)
         RES=AMIN1(1.E12,AMAX1(-1.E12,F(I)-FA))
       CHI=CHI+RES**2*WX(I)
       W1=2*(FA-F(I))*WX(I)
       W2=2*WX(I)
         KJ=0
       DO 80 J=1,NT
       JC=3*(J-1)+1
       PC(J)=PC(J)+W1*P(JC)
       W2P=W2*P(JC)
       DO 80 K=1,J
         KC=3*(K-1)+1
         KJ=KJ+1
80     PPCC(KJ)=PPCC(KJ)+P(KC)*W2P
200    CONTINUE
       IF(CHI.LE.1.01*CHL.AND.KIT.GE.NKIT)GOTO 2502
       IF(CHI.LE.1.01*CHL.AND.IEN.EQ.1)GOTO 2502
       IF(CHL-CHB.GT..1D0)GOTO 248
       IF(KIT.LE.3)GOTO 248
         IEN=1
       IF(CHI.LT.CHL)GOTO 2502
248    CONTINUE
       CALL SMSQ(CHI,CHB,CHL,PC,PPCC,AM,FR,CONS,SM,NT,1)
       DO 260 J=1,NSC
260    C(J)=DMAX1(.001D0,CONS(J))
       KIT=KIT+1
       IF(KIT.GT.2*NKIT)GOTO 280
       GOTO 28
2502   CONTINUE


C *** CHANGING SIGN OF SMALL PEAKS
       IF(KSS.LT.10)THEN
       KSS=KSS+1
       KIT=1
         DO 249 I=1,NSC
       IF(C(I)*C(I)*w(I)*WX(I).LT..1D0)IPT(I)=-IPT(I)
       PRINT*,' SIGN CHANGE AT XP',XP(I),C(I)
       CONS(I)=C(I)
249      CONTINUE
       IEN=0
       CHB=1.D32
       CHL=1.D33
       GOTO 28
       ENDIF
```

```
        CHI=CHI-PEN
        DO 2503 I=1,NDT
2503    AM(I)=PPCC(I)
        CALL SMINV(AM,NT,IFL)
        DO 2510 I=1,NDT
2510    AM(I)=DMIN1(1.D9,DMAX1(-1.D9,AM(I)))
        CHIP=DMAX1(1.D0,CHI/MAX0(1,IE-IB+1-NT))
        DO 2520 J=1,NSC
        EC(J)=2*(2*C(J)*W(J))**2*AM(J*(J+1)/2)*CHIP
      EC(J)=DSQRT(DMAX1(1.D-30,EC(J)))
        EW(J)=0
        EX(J)=0
        WF(J)=W(J)
2520    CONTINUE
         IF(IPP.EQ.0)RETURN
        NP=IE-IB+1
      WRITE(*,104)NP
104   FORMAT(' WE ARE FITTING',I5,' POINTS')
      WRITE(*,102)CHI,PEN
102   FORMAT(' CHISQUARE IS',E20.7,' PEN IS',E20.7)
      IF(NSC.GT.0)WRITE(*,103)(I,C(I),EC(I),XP(I),EX(I),W(I),EW(
     # I),I=1,NSC)
103   FORMAT(' THE B SPLINE COEFFS'/'    #',8X,'C(I)',17X,'XP(I)',
     # 18X,'W(I)'/(I5,3(F12.4,F10.6)))
       RETURN
280   WRITE(*,105)CHI,CHB,CHL,(CONS(I),I=1,NT)
105   FORMAT(' LOOPING IN QMIN, CHI,CHB,CHL/CONS'/(3E20.12))
      NPART=1
      DO 290 J=1,NSC
290   C(J)=0
      RETURN
      END

C*********************************************************************
        SUBROUTINE FWHM(NSC,AI,WRAT)
C*********************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
         DIMENSION AI(1),WRAT(1)
      COMMON/SCONS/DU(150),XPS(5),XPL(5),NTYPE
C     TRAP RULE INTEGRATION AND HALF WIDTH FINDING
         DO 80 ITY=1,NTYPE
      XLH=-1.D0
        CALL XHMAX(XLH,ITY)
        XUH=1.D0
        CALL XHMAX(XUH,ITY)
        WRAT(ITY) =XUH-XLH
         H=WRAT(ITY)/50
      X=XPS(ITY)-H/2
      AI(ITY)=0.D0
25      X=X+H
      F=POLYG(X,ITY,SP)
      AI(ITY)=AI(ITY)+F
        IF(X.LT.XPL(ITY))GOTO 25
      AI(ITY)=AI(ITY)*H
      WRITE(*,101)XLH,XUH,AI(ITY)
101   FORMAT(' FWHM --- XLH=',E16.7,' XUH=',E16.7,' AI=',E16.7)
```

```
80      CONTINUE
        RETURN
        END
C*************************************************************************
        SUBROUTINE XHMAX(XT,ITYPE)
C*************************************************************************
        IMPLICIT REAL*8(A-H,O-Z)
C IF XT INITIALLY < 0 THE XT RETURNED WILL BE THE LOWER VALUE
C IF XT INITIALLY > O THE XT RETURNED WILL BE THE UPPER VALUE
        NLOOP=0
        FA=.5
        XA=0
        XB=XT
5       FB=POLYG(XB,ITYPE,SP)-.5
        IF(FB.LT.0.)GOTO 10
        XB=2*XB
        GOTO 5
10      XT=XA-FA*(XA-XB)/(FA-FB)
        FT=POLYG(XT,ITYPE,SF)-.5
        NLOOP=NLOOP+1
        IF(NLOOP.GT.100)GOTO 50
        IF(DABS(FT).LT.1.D-6.OR.DABS(XA-XB).LT.1.D-6)RETURN
        IF(FT.LT.0)GOTO 20
        FA=FT
        XA=XT
        GOTO 10
20      FB=FT
        XB=XT
        GOTO 10
50      WRITE(*,100)NLOOP,XA,XB,XT,FA,FB,FT
100     FORMAT(' XHMAX IN A LOOP, NLOOP,XA,XB,XT,FA,FB,FT'/I5,6E
     # 20.6)
        RETURN
        END


C*************************************************************************
      SUBROUTINE RCPLOT(IB,IE,FB,FI,AB,CN,XN,WN,IPT,P,NS,XOFF)
C*************************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      REAL*4 FB,FI,SQRT,AMAX1,AMIN1,FMIN,FMAX,FA,AB,ARG,XOFF,FSMALL
        INTEGER*2 IDIV,IX,IEND,IRES,IDAT,IFA,IFB,ICH
      DIMENSION P(1),FB(1),FI(1),CN(1),XN(1),
     # WN(1),IPT(1),AB(1),ICH(6)
      EQUIVALENCE(BSMALL,ICH(1))
        DATA IFA,IFB,ICH/8*0/
        WRITE(*,1901)IB,IE
1901    FORMAT(' PCGRAPH IB,IE',2I5)
C *** FIND SMALLEST VALUE OF F TO BIAS UPWARDS
      FSMALL=0
      DO 10 I=IB,IE
10    FSMALL=AMIN1(FSMALL,FI(I))
      BSMALL=FSMALL
        IEND=-2
        IX=IB+XOFF
      IDIV=-100
      IFA=1
      IFB=1
```

```
C        PRINT*,' RCPLOT',IDIV,IX,IFA,IFB,ICH
         WRITE(3)IDIV,IX,IFA,IFB,ICH
1199     FORMAT(10Z4)
       DO 1090 J=IB,IE
       X=J+XOFF
       FA=POLYB(X,P,CN,XN,WN,IPT,NS)+FB(J)
         IRES=100*AMIN1(327.,(FI(J)-FA)*SQRT(AE(J)))
         IFA=1000.*ALOG(AMAX1(1.E-3,FA-FSMALL))
         IFB=1000.*ALOG(AMAX1(1.E-3,FB(J)-FSMALL))
         IDAT=1000.*ALOG(AMAX1(1.E-3,FI(J)-FSMALL))
        IF(NS.EQ.0)GOTO 1085
C *** NOW FOR THE PEAKS
         DO 1070 I=1,6
1070     ICH(I)=-10000
         KP=0
        DO 1080 K=1,NS
         KP=KP+1
         IF(KP.EQ.7)KP=1
        ARG=.5*CN(K)*P(1+3*(K-1))
        ARG=1000.*ALOG(AMAX1(1.E-10,ARG))
         ICH(KP)=AMAX1(1.*ICH(KP),ARG)
1080    CONTINUE
1085     CONTINUE
1902     FORMAT(10I6)
         WRITE(3)IRES,IDAT,IFA,IFB,ICH
1090     CONTINUE
         RETURN
         END
C*******************************************************************************
           FUNCTION POLYA(X,C,XP,W,IPT,NS)
C*******************************************************************************
        IMPLICIT REAL*8(A-H,O-Z)
C THIS ROUTINE CALCULATES POLYA= SUM C(I)*STAN(I,X) USING POLYG FOR STAN
C NOTE THAT BY NOT HAVING XPS OR XPL LARGE, WE ASSUME THAT POLYB HAS BEE
C CALLED BEFORE THE FIRST CALL TO POLYA
       COMMON/SCONS/DU(150),XPS(5),XPL(5)
          SAVE /SCONS/
       DIMENSION C(1),XP(1),W(1),IPT(1)
       POLYA=0
       IF(NS.EQ.0)RETURN
       DO 200 J=1,NS
       XM=X-XP(J)
         ITEST=IPT(J)
       IF(XM.LE.W(J)*XPS(ITEST).OR.XM.GE.W(J)*XPL(ITEST))GOTO 200
       XM=XM/W(J)
C  PARTIAL WRT C(J)
        FADD=C(J)*POLYG(XM,ITEST,SP)
       POLYA=POLYA+C(J)*FADD
200    CONTINUE
       RETURN
       END
C*******************************************************************************
        FUNCTION POLYB(X,P,C,XP,W,IPT,NS)
C*******************************************************************************
        IMPLICIT REAL*8(A-H,O-Z)
C THIS ROUTINE CALCULATES POLYB=SUM CI*SI ALONG WITH DPOLYB/DCI
C THE NEXT NS SI'S, WHICH HAVE DERIVATIVES WITH RESPECT TO C,XP,AND W
C ARE BSPLINES
```

```
         SAVE /SCONS/
        COMMON/SCONS/DU(150),XPS(5),XPL(5)
         DIMENSION P(1),C(1),XP(1),W(1),IPT(1)
        NC=1
5       NPART=1
        POLYB=0
        DO 200 J=1,NS
          ITEST=IPT(J)
        P(NPART)=0.
        P(NPART+1)=0.
        P(NPART+2)=0.
        XM=X-XP(J)
        IF(XM.LE.W(J)*XPS(ITEST).OR.XM.GE.W(J)*XPL(ITEST))GOTO 200
        XM=XM/W(J)
C  PARTIAL WRT C(J)
         FADD=C(J)*POLYG(XM,ITEST,SP)
        P(NPART)=2.*FADD
        POLYB=POLYB+C(J)*FADD
C PARTIAL WRT XP(J)
        P(NPART+1)=-C(J)*C(J)*SP/W(J)
C PARTIAL WRT W(J)
        P(NPART+2)=P(NPART+1)*XM
200     NPART=NPART+3
        RETURN
        END
C*******************************************************************************
        FUNCTION POLYG(X,IPT,SP)
C*******************************************************************************
         IMPLICIT REAL*8(A-H,O-Z)
          CHARACTER*64 NA
        COMMON/SCONS/W(10,5),XP(10,5),C(10,5),XPS(5),XPL(5),IP,NC
C THIS ROUTINE CALCULATES POLYG=SUM CI*SI ALONG WITH DPOLY/DX
C IPT GIVES THE PEAK TYPE RANGING FROM 1 TO 5
          DIMENSION NST(5)
           SAVE
        IF(NC.EQ.1)GOTO 30
        NC=1
          IP=0
3         PRINT*,' ENTER THE NAME OF THE FILE WITH THE STANDARD PEAK'
          READ(2,'(A)')NA
          IF(NA.EQ.'\'.OR.NA.EQ.'STOP')GOTO 30
          OPEN(11,FILE=NA,STATUS='OLD',ERR=3)
        WRITE(*,102)
102     FORMAT(' THE FOLLOWING CONSTANTS,PEAKS,AND WIDTHS ARE THE',
       X ' STANDARD PEAK')
          IP=IP+1
          NS=1
          IF(IP.GT.5)GOTO 300
5       READ(11,101,END=20)C(NS,IP),XP(NS,IP),W(NS,IP)
101     FORMAT(5X,3E20.7)
        WRITE(*,103)NS,C(NS,IP),XP(NS,IP),W(NS,IP)
103     FORMAT(I5,3E20.7)
        NS=NS+1
        GOTO 5
20      NS=NS-1
          CLOSE(11)
        XPS(IP)=1.D32
        XPL(IP)=-1.D32
```

```
      DO 25 J=1,NS
      XPS(IP)=DMIN1(XPS(IP),XP(J,IP)-W(J,IP))
25    XPL(IP)=DMAX1(XPL(IP),XP(J,IP)+W(J,IP))
        NST(IP)=NS
        GOTO 3
30    CONTINUE
       POLYG=0.
      SP=0.
        NS=NST(IPT)
      DO 200 J=1,NS
      XM=X-XP(J,IPT)
      IF(XM.LE.-W(J,IPT).OR.XM.GE.W(J,IPT))GOTO 200
      XM=XM/W(J,IPT)
      FADD=C(J,IPT)*((1.+XM)*(1.-XM))**3
      POLYG=POLYG+FADD
       SP=SP+3.*(FADD/(1.+XM)-FADD/(1.-XM))/W(J,IPT)
200   CONTINUE
      RETURN
300     PRINT*,' ATTEMPT TO DEFINE MORE THAN FIVE STANDARDS'
        STOP
      END

C*************************************************************************
      SUBROUTINE RESL(F,FB,WX,FW,ALR,IQM,ILR,IFTC,IC,N,XOFF,CUTOFF)
C*************************************************************************
      DIMENSION ILRO(50),ISRO(50),F(1),FB(1),WX(1),IWT(5)
         COMMON/SCONS/DU(150),XPS(5),XPL(5),IP,NGPCAL
      REAL*8 FW(3,5),XPS,XPL,DU
         SAVE ILRO,N10,ISRO
        DATA N10/50/
      IFTC=0
      IF(IQM.LE.N10)GOTO 20
15    IFTC=1
      RETURN
20    ALR=0
      XT=XOFF+N/2
      DO 90 J=1,IP
        WPG=DSQRT(DMAX1(3D0,FW(1,J)+XT*(FW(2,J)+XT*FW(3,J))))
      IW=WPG
      IW=MAX0(MIN0(100,IW),4)
      IWT(J)=IW
      JM=J-1
      DO 30 K=1,JM
30    IF(IW.EQ.IWT(K))GOTO 90
        CALL SCAN(F,FB,WX,N,ILRTE,ALRT,IW,ILRO,ISRO,IQM,IC)
      IF(ABS(ALR).GT.ABS(ALRT))GOTO 40
      ILR=ILRTE
      ILRT=J
      ALR=ALRT
40    CONTINUE
90    CONTINUE
      IF(ABS(ALR).LT.CUTOFF.OR.IQM.GT.50)GOTO 15
        IQM=IQM+1
        IF(IQM.LE.50)THEN
         ILRO(IQM)=ILR
         ISRO(IQM)=1
         IF(ALR.LT.0.)ISRO(IQM)=-1
      ENDIF
```

```
      RETURN
      END
C*********************************************************************
        SUBROUTINE SCAN(F,FB,WX,N,ILR,ALR,IW,ILRO,ISRO,IQM,IC)
C*********************************************************************
      DIMENSION FSM(100),ILRO(50),ISRO(50),F(1),FB(1),WX(1)
      IWM=IW-1
      IWS2=IW/2
      DO 10 I=1,IW
10    FSM(I)=0
      SUM=0
      ALR=0
      DO 100 I=1,N
      FSM(IW)=(F(I)-FB(I))*SQRT(ABS(WX(I)))
      SUM=SUM+FSM(IW)-FSM(1)
      DO 20 J=1,IWM
20    FSM(J)=FSM(J+1)
      IF(ABS(ALR).GT.ABS(SUM))GOTO 40
      ILRT=I-IW
      FMAX=-1.E32
      DO 25 J=2,IW
      IF(FSM(J).LT.FMAX)GOTO 25
      FMAX=FSM(J)
      ILRA=ILRT+J
25    CONTINUE
      ILRT=MAX0(1,MIN0(N,ILRA))
      DO 30 J=1,IQM
      IF(ILRT.EQ.ILRO(J))GOTO 40
      IF(SUM*ISRO(J).GT.0..AND.IABS(ILRT-ILRO(J)).LE.IC)GOTO 40
30    CONTINUE
35    CONTINUE
      ALR=SUM
      ILR=ILRT
40    CONTINUE
100   CONTINUE
      ALR=ALR/SQRT(1.*IW)
      RETURN
      END

C*********************************************************************
       SUBROUTINE ROPKS(C,SC,W,IPT,WF,SW,XP,SXP,N,NPP,FW,
     # XOFF,NAI)
C*********************************************************************
        REAL*8 C,SC,W,SW,WF,XP,SXP,FW,DSQRT,STR
        CHARACTER*64 NA,NAI
        DIMENSION C(1),IPT(1),SC(1),W(1),WF(1),SW(1),
     # XP(1),SXP(1),FW(3,5),ARAT(5),WRAT(5)
        DATA ARAT,WRAT/10*1./
C *** THIS ROUTINE IS A FIRST PASS ATTEMPT TO PLACE PEAKS IN
C *** SPECIFIED LOCATIONS -- PEAKS MUST BE ORDERED BY CHANNEL
C *** NUMBER
        NPP=0
        IF(NAI.EQ.'NONE')RETURN
      PRINT*,NAI
        OPEN(10,FILE=NAI,STATUS='OLD',ERR=335)
        WRITE(*,103)NAI
103     FORMAT(' FILE=',A64/' HAD STARTING GUESSES FOR PEAK LOCATIONS')
        READ(10,'(A)')NA
```

```
        WRITE(*,104)NA
        READ(10,'(A4,I5)')NA,IP
        PRINT*,NA,IP
        DO 4 I=1,2
        READ(10,'(A)')NA
4       WRITE(*,104)NA
        IF(IP.NE.0)READ(10,'(10F8.4)')(ARAT(I),WRAT(I),I=1,IP)
104     FORMAT(1X,A64)
5       READ(10,*,END=150)XP(NPP+1),DUMM,W(NPP+1),DUM2,STR,DUM3
     # ,IPT(NPP+1)
        IF(XP(NPP+1)-XOFF.LT.0.)GOTO 5
        IF(XP(NPP+1)-XOFF.GT.N)GOTO 150
        NPP=NPP+1
      K=IABS(IPT(NPP))
      ARG=FW(1,K)+XP(NPP)*(FW(2,K)+XP(NPP)*FW(3,K))
      ARG=DMAX1(.5D0,ARG)
      WF(NPP)=DSQRT(ARG)
        IF(W(NPP).NE.0.D0)WF(NPP)=W(NPP)
        SW(NPP)=.99D0
        W(NPP)=WF(NPP)
        IRAT=IABS(IPT(NPP))
        STR=STR/ARAT(IRAT)
        W(NPP)=W(NPP)/WRAT(IRAT)
        IF(STR.GT.0)THEN
        C(NPP)=DSQRT(STR/W(NPP))
        IPT(NPP)=IABS(IPT(NPP))
      ELSE
        C(NPP)=-DSQRT(-STR/W(NPP))
        IPT(NPP)=-IABS(IPT(NPP))
      ENDIF
        GOTO 5
150     CLOSE(10)
235     RETURN
335     PRINT*,' COULD NOT OPEN FILE ',NAI
      READ(*,*)ITEST
      STOP
        END
C*********************************************************************
      SUBROUTINE SMSQ(CHI,CHB,CHL,PC,PPCC,A,FR,CONS,SM,NT,NW)
C*********************************************************************
        IMPLICIT REAL*8 (A-H,O-Z)
C *** PPCC AND A MUST BE REAL*8.  THE OTHER ARGUMENTS CAN BE
C *** DECLARED REAL*4 AND THE ROUTINE WILL STILL WORK
C THE ROUTINE FINDS THE NECESSARY CHANGES IN CONS FOR THE NEW
C VALUE OF CHI TO BE EQUAL TO FR*CHI
C CHI IS THE CURRENT VALUE OF CHI (THE QUANTITIY BEING MINIMIZED)
C CHB IS THE VALUE PREDICTED FOR CHI ON THE LAST CALL TO SMSQ
C CHL IS THE LAST VALUE OF CHI, SM IS A VECTOR CONTAINING THE
C RELATIVE SMOOTHING FOR EACH CONSTANT IN THE VECTOR CONS.
C PC IS THE SET OF FIRST DERIVATIVES OF CHI WITH RESPECT TO THE CONSTANT
C PPCC IS THE SET OF SECOND DERIVS (PPCC(I+J*(J-1)/2) FOR I < J)
C STORED IN PACKED FORM, SEE SMINV
C NOTE THAT WHEN CHB=CHL, THE ROUTINE CAN GET NO LOWER
C NT IS THE NUMBER OF CONSTANTS.  PPCC AND AM ARE UPPER TRIANGLE
C  NW DECIDES WHAT TO WRITE
        SAVE NTO,SPC,SPPCC,SCONS,AS
      DIMENSION CONS(1),SM(1),PC(1),PPCC(1),A(1)
     # ,BB(107),B(107),SPC(107),SPPCC(5778),SCONS(107),PPCCD(107)
```

```
C *** THE ABOVE MUST BE DIMENSIONED NT OR LARGER
      DATA X0,X1,X2,AS/3*0.D0,-1.E0/
      DATA NTO/0/,TOL/1.D-6/
        NDT=NT*(NT+1)/2
      IFL=0
      IRT=0
      IAB=0
      QB=1.D0
      ILEFT=0
      ICYCLE=0
       IAA=0
        IF(NT.GT.107)GO TO 2500
      IF(NT.NE.NTO)THEN
      NTO=NT
        CHB=1.D33
        CHL=1.D34
        ENDIF
      AS=DMAX1(DMIN1(AS+2,75D0),-37D0)
C SAVE THE LAST SET OF COEFFS IF CHI<CHL
      IF(NW.GE.1)WRITE(*,'(A,3G20.7)')' CHI,CHB,CHL',CHI,CHB,CHL
      IF(CHI.GT.CHL)GO TO 10
      IF(1.000001*CHI.LT.CHL)IR=0
        DO 8 I=1,NT
        BB(I)=CONS(I)
        SCONS(I)=CONS(I)
8       SPC(I)=PC(I)
        DO 9 I=1,NDT
9       SPPCC(I)=PPCC(I)
      IF((CHI-CHB)/(CHL-CHI+1.E-6).GT..1)FR=.1*(9*FR+1)
      IF((CHI-CHB)/(CHL-CHI+1.E-6).GT..5)GOTO 20
      FR=DMAX1(1.D-2,FR*FR)
      GOTO 20
10    CONTINUE
C RESET TO THE BEST COEFFS (BEWARE THE ROUTINE MUST HAVE RUN BEFORE)
      AS=DMIN1(AS+17.D0,75D0)
      FR=.25*(3.+CHB/CHL)
       IR=IR+1
       CHI=CHL
         DO 18 I=1,NT
        CONS(I)=SCONS(I)
        BB(I)=CONS(I)
18      PC(I)=SPC(I)
        DO 19 I=1,NDT
19      PPCC(I)=SPPCC(I)
      IF(NW.GE.1)WRITE(*,191)IR
191   FORMAT(' IR=',I5)
      IF(IR.GT.20)THEN
        CHB=CHI
        RETURN
        ENDIF
20    CONTINUE
      IF(NW.GE.2)WRITE(*,173)FR
173   FORMAT(' FR=',E10.3)
      CHB=CHI
25    DE=DEXP(AS)
        KI=0
      DO 32 I=1,NT
        KI=KI+I
```

```
      PPCCD(I)=PPCC(KI)
32    PPCC(KI)=PPCC(KI)+SM(I)*DE
38      CONTINUE
        CALL GSOLVE(PPCC,A,PC,B,NT)
        KI=0
        DO 40 I=1,NT
        KI=KI+I
40      PPCC(KI)=PPCCD(I)
      ICYCLE=ICYCLE+1
C *** NOTE THAT B HERE IS THE SAME AS X IN GSOLVE
52    AJP=CHI
      IF(NW.GE.4)WRITE(*,10017)(B(I),I=1,NT)
        KI=0
      DO 60 K=1,NT
        KI=KI+K
      IF(DABS(B(K)).GT.1.D10)B(K)=DSIGN(1.D10,B(K))
      AJP=AJP+B(K)*(PC(K)+0.5*PPCC(KI)*B(K))
      IF(K.EQ.1)GOTO 60
      KM=K-1
        KT=KI-K
      DO 55 M=1,KM
        KT=KT+1
55    AJP=AJP+PPCC(KT)*B(M)*B(K)
60    CONTINUE
      IF(NW.GE.3)WRITE(*,104)AJP
104   FORMAT(' AJP=',E20.12)
C ****** AJP WAS CALCULATED ABOVE, WILL BE TESTED BELOW *************
C **** IF WE CAN FIND AJP < FR*CHI, WE CAN INTERPOLATE IN AS TO
C **** FIND A VALUE FOR WHICH AJP=FR*CHI, IFL = 1 INDICATES THIS CONDITI
      DT=AJP/DMAX1(1.D-35,CHI)
      IF(NW.GE.3)WRITE(*,105)DT
105   FORMAT(' DT=',E20.12)
      IF(DT.GE.FR)IAB=1
      IF(DT.LT.FR)IFL=1
      IF(DT-1.E0.LT.1.E-7)GOTO 1000
      IF(NW.GE.1)WRITE(*,109)AJP,AS
109   FORMAT(' THE MATRIX INVERSION APPEARS WRONG, AJP,AS',2E15.6)
C *********** THIS INVERSION IS WRONG, BUT A PREVIOUS ONE WAS RIGHT THU
        IF(NW.GE.3)WRITE(*,3421)IAA,AS
3421    FORMAT(' IAA,AS',I5,E20.6)
      IF(IAA.EQ.1)GOTO 1000
C ********** THIS INVERSION IS WRONG AND WE HAVE NEVER HAD A CORRECT ONE
      IF(AS.GE.76.)THEN
        IR=I+1
        GOTO 10
        ENDIF
115   AS=AS+3.
      GOTO 25
120   CONTINUE
C FINDING X'S ON EACH SIDE OF FR*CHI
      IF(ICYCLE.GT.50)AS=DMIN1(75.D0,AS+30.E0)
      IF(ICYCLE.GT.30.OR.DABS((AJP-FR*CHI)/CHI).LT.TOL)GOTO 3117
      IF(AJP.GT.FR*CHI)GOTO 130
      IRT=IRT+1
       ILEFT=0
      AR=AS
      QR=AJP/CHI-FR
      GOTO 140
```

```
130   AL=AS
      QL=AJP/CHI-FR
      IRT=0
      ILEFT=ILEFT+1
140   X0=X1
      X1=X2
      IF(IFL*IAB.EQ.0)GOTO 1150
      IF(QB.GT.1.-FR)GOTO 115
      IF(DABS((AR-AL)).LT..1E-5)GOTO 3117
      AS=AL-QL*(AR-AL)/(QR-QL)
      X2=AS
      IF(NW.GE.2)WRITE(*,102)AL,QL,AR,QR,AS
102   FORMAT(' AL,QL',2E15.6,' AR,QR',2E15.6,' AS=',E15.6)
      IF(IRT.LT.2.AND.ILEFT.LT.2)GOTO 25
C AITKENS EXTRAPOLATION
      ALPHA=(X1**2-X0*X2)*(2.*X1-X2-X0)/((2.*X1-X2-X0)**2+1.E-37)
      IF(ALPHA.LT.AL.AND.AL.LT.AR)ALPHA=.5*(AL+AR)
      IF(ALPHA.GT.AL.AND.AL.GT.AR)ALPHA=.5*(AL+AR)
      IF(ALPHA.LT.AR.AND.AR.LT.AL)ALPHA=.5*(AL+AR)
      IF(ALPHA.GT.AR.AND.AR.GT.AL)ALPHA=.5*(AL+AR)
      IF(DABS((ALPHA-AR)/(AR-AL)).LT..1E-1.OR.DABS((ALPHA-AL)/(AR-AL))
     # .LT..1E-1)ALPHA=.5*(AL+AR)
      AS=ALPHA
      IF(NW.GE.2)WRITE(*,103)AS
103   FORMAT(' AITKENS EXTRAPOLATION, AS',E15.6)
      ILEFT=0
      IRT=0
      GOTO 25
C ********* WE HAVE FOUND 0 < AJP < CHI ***********************
C ********* AND WE WANT TO SAVE THE BEST SET OF B'S ***********
1000  QC=DABS(AJP/CHI-FR)
      IF(IAA.EQ.0)GOTO 1050
      IF(QC.GT.QB)GOTO 1120
1050  CHB=AJP
      IAA=1
      QB=QC
        ASB=AS
      IF(NW.GE.2)WRITE(*,117)QB,AJP,AS
117   FORMAT(' QB,AJP,AS',3E15.6)
      DO 1115 I=1,NT
1115  BB(I)=B(I)
      IF(QB.LT.TOL)GOTO 3117
1120  GOTO 120
C ********* TRY AGAIN WITH MORE OR LESS SMOOTHING **********
1150  AS=AS-4.0
        IF(IR.GT.1)AS=.5*(AS+ASB)
      IF(IAB.EQ.0)AS=AS+8.
      IF(AS.LE.-75.)GOTO 3117
      IF(AS.GE.76.)GOTO 3117
      GOTO 25
C ****************** EXITING WITH THE BEST COEFF'S FOUND ***********
3117  CONTINUE
      CHL=CHI
      DO 3119 I=1,NT
3119  CONS(I)=CONS(I)+BB(I)
      IF(NW.GE.3)WRITE(*,10017)(BB(I),I=1,NT)
10017 FORMAT(1X,6E13.6)
      IF(NW.GE.3)WRITE(*,10018)QB
```

```
10018 FORMAT(' THE FINAL QB IS',E20.12)
        IF(NW.GE.1)PRINT'(A,G11.4)',' FINAL AS IS',AS
      RETURN
2500    WRITE(*,4367)NT
4367    FORMAT(' NT OF',I5,' IS TOO LARGE FOR THE'/
     # ' DIMENSION OF B AND BB AND ALSO TERMS IN GSOLVE')
        STOP
      END
C*********************************************************************
        SUBROUTINE SMINV(AP,N,IFL)
C*********************************************************************
        IMPLICIT REAL*8 (A-H,O-Z)
        DIMENSION AP(1)
C *** THIS ROUTINE BEGINS IN LINPACK CHAPTER 3 - POSITIVE
C *** DEFINITE SYMMETRIC MATRICES
C *** IN OUR CASE (CURVE FITTING) WE CAN ALWAYS WRITE THE APPROXIMATE
C*********************************************************************
C *** FUNCTION AS SUM C*SJ(XI).  IF WE THEN FORM X**2 = SUM ON I
C*********************************************************************
C *** OF SUM ON J (CJ*SJ(XI))**2 WE FIND X**2 = SUM ON K CK *
C *** SUM ON J CJ * AJK WHERE AJK IS SUM ON I SJ(XI)*SK(XJ), THE
C *** POSITIVE DEFINITE SYMMETRIC MATRIX OF INTEREST HERE
C
C *** AP IS A PACKED MATRIX AS IN THE FOLLOWING
C *** K=0
C     DO 20 J=1,N
C     DO 20 I=1,J
C     K=K+1
C     AP(K)=A(I,J)
C 20  CONTINUE
        IFL=0
        JJ=0
        DO 30 J=1,N
        S=0
        KJ=JJ
        KK=0
        IF(J.LT.2)GOTO 20
        JM1=J-1
        DO 10 K=1,JM1
        KJ=KJ+1
        T=AP(KJ)
        KM1=K-1
        IF(KM1.LT.1)GOTO 8
        DO 5 I=1,KM1
5       T=T-AP(I+KK)*AP(I+JJ)
8       KK=KK+K
        T=T/AP(KK)
        AP(KJ)=T
        S=S+T*T
10      CONTINUE
20      CONTINUE
        JJ=JJ+J
        SMT=DMAX1(S,AP(JJ))
        DIFF=AP(JJ)-S
        IF(DIFF.GT.1.D-10*SMT.AND.DIFF.GT.1.D-70)GOTO 25
C        WRITE(*,1975)J,AP(JJ),S
1975    FORMAT(' POS DEF QUANT LE 0. J=',I5,' AP(JJ)=',E10.3,
     # ' S=',E10.3)
```

```
C *** FIX IS MORE SMOOTHING FOR A(J,J)
        IFL=-1
        DIFF=1.D30
25      AP(JJ)=DSQRT(DIFF)
30      CONTINUE
C *** NEXT WE CONSTRUCT THE INVERSE MATRIX
        KK=0
        DO 100 K=1,N
        K1=KK
        KK=KK+K
        AP(KK)=1/AP(KK)
        T=-AP(KK)
        KM=K-1
        IF(KM.LE.0)GOTO 86
        DO 85 I=1,KM
85      AP(I+K1)=T*AP(I+K1)
86      KP1=K+1
        J1=KK
        KJ=KK+K
        IF(N.LT.KP1)GOTO 90
        DO 80 J=KP1,N
        T=AP(KJ)
        AP(KJ)=0
        DO 70 I=1,K
70      AP(J1+I)=AP(J1+I)+T*AP(K1+I)
        J1=J1+J
        KJ=KJ+J
80      CONTINUE
90      CONTINUE
100     CONTINUE
        JJ=0
        DO 130 J=1,N
        J1=JJ
        JJ=JJ+J
        JM1=J-1
        K1=0
        KJ=J1+1
        IF(JM1.LT.1)GOTO 120
        DO 110 K=1,JM1
        T=AP(KJ)
        DO 105 I=1,K
105     AP(I+K1)=AP(I+K1)+T*AP(I+J1)
        K1=K1+K
        KJ=KJ+1
110     CONTINUE
120     CONTINUE
        T=AP(JJ)
        DO 125 I=1,J
125     AP(I+J1)=     T*AP(I+J1)
C ABOVE LINE ORIGINALLY HAD ANOV
130     CONTINUE
140     CONTINUE
        RETURN
        END
C*************************************************************************
        FUNCTION ANOV(X,Y)
C*************************************************************************
        IMPLICIT REAL*8 (A-H,O-Z)
```

```
      EQUIVALENCE (IX,XT),(IY,YT)
C     ANOV='FFFF7CFF'X*DSIGN(1.D0,X)*DSIGN(1.D0,Y)
      XT=X
      YT=Y
C     IF((IX.AND.'7F80'X)+(IY.AND.'7F80'X).LT.'BC80'X)
C    #  ANOV=XT*YT
       ANOV=XT*YT
      RETURN
      END
C*********************************************************************
 SUBROUTINE GSOLVE(A,AM,B,X,NT)
C*********************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
C *** SOLVES AX=-B IN CASES WHERE THE MATRIX CAN BE INVERTED CHEAPLY
C *** AND ACCURATELY WITH NO PROBLEMS
      DIMENSION A(1),AM(1),B(1),X(1)
      NTD=NT*(NT+1)/2
      DO 10 I=1,NTD
10    AM(I)=A(I)
      CALL SMINV(AM,NT,IFL)
      CALL SMMULT(AM,B,X,NT)
      DO 20 I=1,NT
20    X(I)=-X(I)
      RETURN
      END
C*********************************************************************
      SUBROUTINE SMMULT(A,B,C,N)
C*********************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION A(1),B(1),C(1)
C *** FOR PACKED MATRICES C(N)=A(N*(N+1)/2)*B(N)
      DO 10 I=1,N
10    C(I)=0
      K=0
      DO 20 I=1,N
      IM=I-1
      DO 15 J=1,IM
      K=K+1
      C(I)=C(I)+A(K)*B(J)
15    C(J)=C(J)+A(K)*B(I)
      K=K+1
20    C(I)=C(I)+A(K)*B(I)
      RETURN
      END
C*********************************************************************
      FUNCTION KIJ(I1,I2)
C*********************************************************************
C      WRITE(*,'(A,2I5)')' IN KIJ I1,I2',I1,I2
      I=MIN0(I1,I2)
      J=MAX0(I1,I2)
      KIJ=(J*(J-1)/2)+I
      RETURN
      END
C*********************************************************************
       FUNCTION CFEST(W,ITT,F,FB,WX,XT,N,ERRSQ,IFW)
C*********************************************************************
       IMPLICIT REAL*8 (A-H,O-Z)
      CHARACTER*4 IFW
```

```
      REAL*4 F(1),FB(1),WX(1)
      NIT=0
      IBG=XT-2*W
      IBG=MAX0(1,IBG)
      IEND=XT+W
      IEND=MIN0(N,IEND)
      ANUM=0
      ADEN=0
     ERRT=0
     ERRSQ=0
      DO 10 I=IBG,IEND
      X=(I-XT)/W
      DF=(F(I)-FB(I))
     ERRSQ=ERRSQ+DF*DF*WX(I)
      FA=DMAX1(1.D-4,POLYG(X,IABS(ITT),SP) )
      ANUM=ANUM+DF*FA*WX(I)
      ADEN=ADEN+FA*FA*WX(I)
10    CONTINUE
      CFEST=0
      IF(ADEN.LE.0.)RETURN
      CFEST=ANUM/ADEN
     ERRSQ=ERRSQ-CFEST*ANUM
     ERRSQ=ERRSQ/(IEND+1-IBG)
     IF(CFEST.GT.0)THEN
        CFEST=DSQRT(CFEST)
       ITT=IABS(ITT)
       RETURN
     ELSE
       IF(IFW.EQ.'FIXN')THEN
         CFEST=DSQRT(-CFEST)
         ITT=-1
         RETURN
       ELSE
         CFEST=0
         RETURN
       ENDIF
     ENDIF
       RETURN
       END
```

The following is a listing of the standard generating code.

```
      EXTERNAL GAUSS,FLOREN,EXALG,ALICOS
      REAL*8 CHI
        CHARACTER*64 NA
        CHARACTER*1 IAN,NO
        CHARACTER*4 INT1,INT,IGAUSS,ILOREN,ZERO,BKG,VOIGHT,LCDAT,HDAT
        CHARACTER*4 IDAT,Z4DAT
        COMMON/DATA/XI(4096),FI(4096),WX(4096),NP
        DIMENSION IHD(4096)
        COMMON/KNOTS/W(10),XP(10),C(10),NSC,NB,CB(4),XB
        COMMON/PASS/ETA,XSUP(100),YSUP(100),NPSUP
        EQUIVALENCE (IHD(1),FI(1))
        CHARACTER*4 II(40),EXTWT
      CHARACTER*64 NADAT
        DATA IGAUSS,ILOREN,NO,ZERO/'GAUS','LORE','N','ZERO'/
        DATA BKG,VOIGHT/'BKGR','VOIG'/,LCDAT/'LCDA'/,HDAT/'HDAT'/
        DATA IDAT/'IDAT'/,Z4DAT/'Z4DA'/,EXTWT/'NORM'/
C *** FLOREN IS A LORENTZIAN, GAUSS IS A GAUSSIAN
106     FORMAT(A64)
        WRITE(*,107)
107     FORMAT(' ENTER THE FILE FOR THE STANDARD PEAK PARAMETERS')
        READ(*,106)NA
        OPEN(12,FILE=NA,STATUS='NEW')
        WRITE(*,103)
103     FORMAT(' ENTER GAUSS, LORENTZ, OR VOIGHT FOR A FIT TO',
     # ' THE APPROPRIATE ANALYTIC PEAK OR BLANK TO FIT DATA')
        READ(*,101)INT1
101      FORMAT(A4)
        WRITE(*,1957)
1957    FORMAT(' ENTER THE NUMBER OF BKG COEF, NUMBER OF SPLINES')
        READ(*,*)NB,NS
        IF(NB.EQ.0)CB(1)=0
        IF(INT1.EQ.IGAUSS.OR.INT1.EQ.ILOREN.OR.INT1.EQ.VOIGHT)GOTO 60
C *** NB IS THE NUMBER OF COEFFICIENTS IN THE BACKGROUND POLY
C *** NS IS THE MAXIMUM NUMBER OF SPLINES USED TO FIT THE SHAPE
      PRINT*,' ENTER THE NAME OF THE DATA FILE'
      READ(*,'(A)')NADAT
        CALL BREAD(II,NADAT,EXTWT)
60       II(1)=INT1
         II(2)='    '
        IF(INT1.EQ.VOIGHT)GOTO 90
        IF(INT1.EQ.IGAUSS)GOTO 80
        IF(INT1.EQ.ILOREN)GOTO 70
        GOTO 290
70      CALL BLI(XI,FI,-32.E0,32.E0,NP,200,FLOREN,1)
        GOTO 290
80      CALL BLI(XI,FI,-3.E0,3.E0,NP,200,GAUSS,1)
        GOTO 290
90      WRITE(*,123)
123     FORMAT(' WHAT VALUE FOR ETA?')
        READ(*,*)ETA
        RANGE=(2/ETA**2)*(-1+SQRT(1+85*ETA**2))
        WRITE(*,124)RANGE
124     FORMAT(' THE RANGE OF X IS 0 TO',F10.5)
        CALL BLI(XSUP,YSUP,0.E0,RANGE,NPSUP,100,EXALG,0)
      PRINT*,'AFTER FIRST BLI CALL'
        CALL BLI(XI,FI,-32.E0,32.E0,NP,200,ALICOS,1)
      PRINT*,'AFTER SECOND BLI CALL'
290     XB=XI(1)
```

```
        NSC=1
300    CALL QMIN(NS,CHI,EXTWT)
        CALL PLOT(II,CHI,EXTWT)
        WRITE(*,125)INT1
125     FORMAT(' THE CONSTANTS ARE FOR ',A4)
        IF(INT1.EQ.VOIGHT)WRITE(*,126)ETA
126     FORMAT(' THE ETA PARAMETER IS ',F10.5)
        CALL LMAX
        STOP
        END
C*******************************************************************************
        SUBROUTINE LCHEX(F,NF,XE)
C*******************************************************************************
        DIMENSION F(1),IH(6)
        CHARACTER*1 HC(16),HI(64)
        DATA HC/'0','1','2','3','4','5','6','7','8',
     # '9','A','B','C','D','E','F'/
        READ(8,*)NMIN,NMAX
        WRITE(*,109)NMIN,NMAX
109     FORMAT(' NMIN, NMAX',2I6)
        NOPE=0
        NF=1
        N1=1
        N2=32
        DO 185 ISKIP=1,8
185     READ(8,1994)
10      READ(8,1994,END=50)(HI(I),I=N1,N2)
        N1=1
1994    FORMAT(9X,32A1)
11      IF(NOPE.LT.NMIN)GOTO 17
        DO 15 L=1,6
        J=L+N1-1
        DO 12 K=1,16
        IF(HI(J).EQ.HC(K))GOTO 15
12      CONTINUE
15      IH(L)=K-1
        IF(NF.GE.1)F(NF)=IH(2)+16*IH(1)+256*(IH(4)+16*IH(3)+256*
     # (IH(6)+16*IH(5)))
        NF=NF+1
17      NOPE=NOPE+1
        IF(NOPE.GT.NMAX)GOTO 50
        N1=N1+6
        IF(N1+5.LT.N2)GOTO 11
        NT=N2-N1+1
        DO 20 I=1,NT
20      HI(I)=HI(I+N1-1)
        N1=NT+1
        N2=N1+31
        GOTO 10
50      CONTINUE
        XE=NOPE-1
60      WRITE(*,112)XE,NOPE
112     FORMAT(' XE, NOPE',F7.0,I6)
        IF(F(NF-1).NE.0)RETURN
        NOPE=NOPE-1
        NF=NF-1
        XE=XE-1
        GOTO 60
```

```
      END
C*********************************************************************
      SUBROUTINE QMIN(NS,CHI,EXTWT)
C*********************************************************************
      COMMON/DATA/XI(4096),FI(4096),WX(4096),NP
      REAL*8 CHI,CHB,CHL,PC,PPCC,A,FR,CONS,SM
     CHARACTER*4 EXTWT
      DIMENSION P(34),CONS(34),SM(34),PC(34),PPCC(595),FRES(4096)
     # ,AM(595),EC(10),EXP(10),EW(10)
      COMMON/KNOTS/W(10),XP(10),C(10),NSC,NB,CB(4),XB
C *** ASSUME THE XI'S ARE IN INCREASING ORDER AND DETERMINE
C *** AN APPROPRIATE PENALTY TERM
      NIB=NP/4
      NIE=(3*NP)/4
      ASP=2*ABS(XI(NIB)-XI(NIE))/NP
      WRITE(*,1932)NIB,NIE,ASP
1932  FORMAT(' NIB,NIE,ASP',2I5,E15.6)
      IF(NSC.GT.1)GOTO 16
      DO 14 J=2,NB
14    CB(J)=0.E0
      CB(1)=AMIN1(FI(1),FI(NP))
      IF(NB.EQ.1)GOTO 15
      CB(1)=FI(1)
      CB(2)=(FI(NP)-FI(1))/NP
15    CONTINUE
      DO 1501 I=1,NP
     IF(EXTWT.EQ.'NORM')WX(I)=1./SQRT(AMAX1(1.,FI(I)))
1501  FRES(I)=(FI(I)-CB(1)-CB(2)*(XI(I)-XB))*WX(I)
     CALL RESIDL(XI,FI,FRES,NP,C(1),XP(1),W(1))
     W(1)=AMAX1(ASP,W(1))
16    IF(NSC.GT.1)NSC=NSC-1
      DO 18 J=1,NB
      CONS(J)=CB(J)
18    SM(J)=10.**(J-1)
23    CONTINUE
      IE=0
      CHB=1.E31
      CHL=1.E32
      FR=0.95
      NT=NB+3*NSC
      NPART=NB+1
      DO 24 J=1,NSC
      CONS(NPART)=C(J)
      SM(NPART)=1.E-2
      CONS(NPART+1)=XP(J)
      SM(NPART+1)=SM(NPART)*1.D6
      CONS(NPART+2)=W(J)
      SM(NPART+2)=SM(NPART)*1.D4
24    NPART=NPART+3
      NKIT=30
      DO 2500 KIT=1,NKIT
      IF(KIT.EQ.NKIT)CHB=CHL
      CHI=0.
      K=0
      DO 30 J=1,NT
      PC(J)=0.
      DO 30 I=1,J
      K=K+1
```

```
30      PPCC(K)=0.
          DO 200 I=1,NP
          FA=POLY(XI(I),P)
          FRES(I)=FI(I)-FA
        IF(EXTWT.EQ.'NORM')WX(I)=1./SQRT(AMAX1(1.,FA))
          FRES(I)=FRES(I)*WX(I)
          CHI=CHI+FRES(I)**2
        W1=-2*FRES(I)*WX(I)
        W2=2*WX(I)*WX(I)
          L=0
          DO 80 J=1,NT
          PC(J)=PC(J)+W1*P(J)
          W2P=W2*P(J)
          DO 80 K=1,J
          L=L+1
80      PPCC(L)=PPCC(L)+P(K)*W2P
200     CONTINUE
          NPART=NB+3
          PEN=0.E0
          DO 210 J=1,NSC
          IF(W(J).GE.ASP)GOTO 210
          PT=(ASP-W(J))**3
          PEN=PEN+PT
          CHI=CHI+PT
          PC(NPART)=PC(NPART)-3.E0*(ASP-W(J))**2
          L=NPART*(NPART+1)/2
          PPCC(L)=PPCC(L)+6.*(ASP-W(J))
210     NPART=NPART+3
          IF(CHL-CHB.GT..1E-1)GOTO 248
          IF(KIT.LE.3)GOTO 248
          IF(IE.EQ.1)GOTO 2502
          IE=1
          IF(CHI.LT.CHL)GOTO 2502
248     CALL SMSQ(CHI,CHB,CHL,PC,PPCC,AM,FR,CONS,SM,NT,1)
          DO 250 J=1,NB
250     CB(J)=CONS(J)
          NPART=NB+1
          DO 260 J=1,NSC
          C(J)=CONS(NPART)
          XP(J)=CONS(NPART+1)
          W(J)=ABS(CONS(NPART+2))
260     NPART=NPART+3
2500     CONTINUE
2502    CONTINUE
          IF(NSC.GE.NS)GOTO 3000
          CALL RESIDL(XI,FI,FRES,NP,C(NSC+1),XP(NSC+1),W(NSC+1))
          W(NSC+1)=AMAX1(ASP,W(NSC+1))
3000    CONTINUE
          WRITE(*,104)NP
104     FORMAT(' WE ARE FITTING',I5,' POINTS')
          CHI=CHI-PEN
          WRITE(*,102)CHI,(I,CB(I),I=1,NB)
102     FORMAT(' CHISQUARE IS',E20.6/' THE BKGRD CONS ARE'/(I5,E20.6))
          IF(NSC.GT.0)WRITE(*,103)(I,C(I),XP(I),W(I),I=1,NSC)
103     FORMAT(' THE B SPLINE COEFFS'/'    #',8X,'C(I)',17X,'XP(I)',
     # 18X,'W(I)'/(I5,3E22.8))
          CHIT=CHI-NP+NB+3*NSC
C         IF(CHIT.LT.0.D0)RETURN
```

```
      IF(NSC.GE.NS)GOTO 3200
      NSC=NSC+1
      IF(NSC.LE.NS)GOTO 23
3200  NSC=NS
        RETURN
        END
C*************************************************************************
       FUNCTION POLY(X,P)
C*************************************************************************
C THIS ROUTINE CALCULATES POLY=SUM CI*SI ALONG WITH DPOLY/DCI
C THE FIRST NB SI'S ARE THE BACKGROUND POLYNOMIAL
C THE NEXT NS SI'S, WHICH HAVE DERIVATIVES WITH RESPECT TO C,XP,AND W
C ARE BSPLINES
        COMMON/KNOTS/W(10),XP(10),C(10),NS,NB,CB(4),XB
        DIMENSION P(40)
        POLY=CB(1)
        IF(NB.EQ.0)POLY=0
        P(1)=1.
        IF(NB.LT.2)GOTO 120
        DO 100 J=2,NB
        P(J)=(X-XB)**(J-1)
100   POLY=POLY+CB(J)*P(J)
120   NPART=NB+1
        DO 200 J=1,NS
        P(NPART)=0.
        P(NPART+1)=0.
        P(NPART+2)=0.
        XM=X-XP(J)
        IF(XM.LE.-W(J).OR.XM.GE.W(J))GOTO 200
        XM=XM/W(J)
C  PARTIAL WRT C(J)
        P(NPART)=2.*C(J)*((1.+XM)*(1.-XM))**3
        FADD=.5*C(J)*P(NPART)
        POLY=POLY+FADD
C PARTIAL WRT XP(J)
        P(NPART+1)=-3.*(FADD/(1.+XM)-FADD/(1.-XM))/W(J)
C PARTIAL WRT W(J)
        P(NPART+2)=P(NPART+1)*XM
200   NPART=NPART+3
        RETURN
        END
C*************************************************************************
      SUBROUTINE LMAX
C*************************************************************************
        CHARACTER*1 IA(80),IBL,IP
        COMMON/KNOTS/W(10),XP(10),C(10),NS,NB,CB(4),XB
        DIMENSION CS(10),ISS(5)
        DATA IBL,IP/' ','+'/
C     FIND MAX X AND MIN X OF PEAK
        XPS=1.D32
        XPL=-1.D32
        DO 30 J=1,NS
        XPS=AMIN1(XPS,XP(J)-W(J))
30    XPL=AMAX1(XPL,XP(J)+W(J))
31    II=0
        H=(XPL-XPS)/200.
        X=XPS+H/2.
33    CONTINUE
```

```
        DO 40 I=1,200
         F=POLYF(X)
         IF(FB.GE.F)GOTO 40
         FB=F
         XB=X
40     X=X+H
         II=II+1
         IF(II.GE.4)GOTO 50
         X=XB-H
         H=H*.01
         GOTO 33
50     CONTINUE
         DO 55 I=1,NS
         C(I)=C(I)/SQRT(FB)
         XP(I)=XP(I)-XB
         CS(I)=C(I)**2
55     CONTINUE
         XPS=XPS-XB
         XPL=XPL-XB
C FINDING THE HALF MAXIMUM POINTS
         XLH=-1
         CALL XHMAX(XLH)
         XUH=1
         CALL XHMAX(XUH)
         WRITE(*,105)XLH,XUH
105    FORMAT(' THE VALUES FOR WHICH F IS .5 ARE',2E20.12)
         AM=1./(XUH-XLH)
         XPS=XPS*AM
         XPL=XPL*AM
         DO 65 I=1,NS
         XP(I)=XP(I)*AM
65     W(I)=W(I)*AM
         WRITE(*,106)XPS,XPL
106    FORMAT(' THE STANDARD COVERS THE INTERVAL',2E20.12)
         WRITE(*,103)
103    FORMAT(' THE FOLLOWING ARE THE CONSTANTS READY FOR RLFIT')
         WRITE(12,101)(I,CS(I),XP(I),W(I),I=1,NS)
101    FORMAT(I5,3E20.7)
         WRITE(*,101)(I,CS(I),XP(I),W(I),I=1,NS)
C GRAPHING THE STANDARD PEAK
         H=(XPL-XPS)/50.
         X=XPS+H/2.
         AI=0.
         IS=-4
         DO 72 I=1,5
         ISS(I)=IS
72     IS=IS+1
         WRITE(*,108)ISS
 108    FORMAT(1H1,3X,5I20)
         DO 80 I=1,50
         F=POLYF(X)
         AI=AI+.02*F
         DO 76 J=1,80
76     IA(J)=IBL
         IF(F.GT.0.E0)       IC=20.*ALOG10(F)+80.0000001E0
         IC=MIN0(80,IC)
         IF(IC.GE.1)IA(IC)=IP
         WRITE(*,102)X,F,IA
```

```
80      X=X+H
102     FORMAT(2E12.4,80A1)
          WRITE(*,107)XB,XB+XLH,XB+XUH,AI
107     FORMAT(' XB=',E14.6,' HALF MAXS AT',2E14.6,' AI=',E14.6)
          RETURN
          END
C*******************************************************************************
          SUBROUTINE XHMAX(XT)
C*******************************************************************************
C IF XT INITIALLY < 0 THE XT RETURNED WILL BE THE LOWER VALUE
C IF XT INITIALLY > O THE XT RETURNED WILL BE THE UPPER VALUE
          NLOOP=0
          FA=.5
          XA=0
          XB=XT
5         FB=POLYF(XB)-.5
          IF(FB.LT.0.)GOTO 10
          XB=2*XB
          GOTO 5
10        XT=XA-FA*(XA-XB)/(FA-FB)
          FT=POLYF(XT)-.5
          NLOOP=NLOOP+1
          IF(NLOOP.GT.10)GOTO 50
          IF(ABS(FT).LT.1.E-6.OR.ABS(XA-XB).LT.1.E-6)RETURN
          IF(FT.LT.0)GOTO 20
          FA=FT
          XA=XT
          GOTO 10
20        FB=FT
          XB=XT
          GOTO 10
50        WRITE(*,100)NLOOP,XA,XB,XT,FA,FB,FT
100       FORMAT(' XHMAX IN A LOOP, NLOOP,XA,XB,XT,FA,FB,FT'/I5,6E
       # 20.6)
          END
C*******************************************************************************
         FUNCTION POLYF(X)
C*******************************************************************************
C THIS ROUTINE CALCULATES POLYF=SUM CI*SI
          COMMON/KNOTS/W(10),XP(10),C(10),NS,NB,CB(4),XB
          POLYF=0.
          DO 200 J=1,NS
          XM=X-XP(J)
          IF(XM.LE.-W(J).OR.XM.GE.W(J))GOTO 200
          XM=XM/W(J)
          FADD=C(J)*C(J)*((1.+XM)*(1.-XM))**3
          POLYF=POLYF+FADD
200     CONTINUE
          RETURN
          END
C*******************************************************************************
          SUBROUTINE PLOT(II,CHIS,EXTWT)
C*******************************************************************************

          REAL*8 CHIS
          COMMON/DATA/XI(4096),FI(4096),WX(4096),NP
          COMMON/KNOTS/W(10),XP(10),C(10),NSC,NB,CB(4),XB
          CHARACTER*4 II(20),EXTWT
```

```
        DIMENSION P(34)
        INTEGER*2 IDIV,IXMUL,IX,IRES,IDAT,IFA,IFB,ICH(6)
        DATA IFA,IFB,ICH/8*0/
      IXMUL=32767/XI(NP)
      IDIV=1
      IFA=1
      FSMALL=0
      IXMUL=32767/XI(NP)
      WRITE(*,1234)IDIV,IFA,IXMUL
1234    FORMAT(' IDIV,IFA,IXMUL',3I10)
        OPEN(3,FILE='SHAPE.GR',STATUS='NEW',FORM='UNFORMATTED')
        WRITE(3)IDIV,IXMUL,(II(I),I=1,2),CHIS
      DO 1090 J=1,NP
      FB=0
      IF(NB.GT.0) FB=CB(1)
      XMXB=XI(J)-XB
      XMXBP=XMXB
        DO 100 I=2,NB
        FB=FB+CB(I)*XMXBP
100   XMXBP=XMXBP*XMXB
      FA=POLY(XI(J),P)
      IF(EXTWT.EQ.'NORM')WX(I)=1./SQRT(AMAX1(1.,FA))
      IRES=AMIN1(32767.,100*(FI(J)-FA)*WX(I))
        IFA=1000.*ALOG(AMAX1(1.E-3,FA-FSMALL))
        IFB=1000.*ALOG(AMAX1(1.E-3,FB-FSMALL))
      IX=AMIN1(32767.,XI(J)*IXMUL)
        IDAT=1000.*ALOG(AMAX1(1.E-3,FI(J)-FSMALL))
      IF(NSC.EQ.0)GOTO 1085
C *** NOW FOR THE PEAKS
        DO 1070 I=1,6
1070    ICH(I)=-10000
        KP=0
        DO 1080 K=1,NSC
        KP=KP+1
        IF(KP.EQ.7)KP=1
        ARG=.5*C(K)*P(NB+1+3*(K-1))
       ARG=1000.*ALOG(AMAX1(1.E-10,ARG))
        ICH(KP)=AMAX1(1.*ICH(KP),ARG)
1080   CONTINUE
1085    CONTINUE
1902    FORMAT(10I6)
        WRITE(3)IRES,IDAT,IFA,IFB,IX,ICH
1090    CONTINUE
        RETURN
        END
C*******************************************************************************
        SUBROUTINE RESIDL(XI,FI,FR,NP,C,XP,W)
C*******************************************************************************
        DIMENSION XI(1),FI(1),FR(1)
        CALL RESL(FR,ALR,JB,NP)
         PRINT*,' ALR,JB,NP',ALR,JB,NP
         XP=XI(JB)
        C=ALR*SQRT(ABS(FI(JB)))
        C=SQRT(ABS(C))
        JS=MAX0(1,JB-2)
        JL=MIN0(NP,JB+2)
50    W=XI(JL)-XI(JS)
        WRITE(*,101)C,XP,W
```

```
101   FORMAT(' IN ZEROTH APPROX C,X,W'/3D12.5)
        RETURN
        END
C*************************************************************************
        SUBROUTINE RESL(FRES,ALR,ILR,N)
C*************************************************************************
        DIMENSION FRES(1)
20      ALR=0
        NM2=N-2
        FSP2=FRES(2)
        FSP1=FRES(1)
        FS=0
        FSM1=0
        FSM2=0
        FSM3=0
        SUM=FSP1+FSP2
      DO 40 I=1,N
        FSM3=FSM2
        FSM2=FSM1
        FSM1=FS
        FS=FSP1
        FSP1=FSP2
        FSP2=0
        IF(I.LT.NM2)FSP2=FRES(I+2)
        SUM=SUM+FSP2-FSM3
      IF(SUM.LT.ALR)GOTO 40
      ILR=I
      ALR=SUM
40    CONTINUE
        ALR=ALR/2.236
      RETURN
      END
C*************************************************************************
      SUBROUTINE BLI(XI,FI,B,E,NP,N,FLOR,NW)
C*************************************************************************
      DIMENSION D(200),XI(1),FI(1)
      PRINT*,'HELLO IN BLI'
101     FORMAT(' IN BLI WITH B=',F8.3,' E=',F8.3,' AND N=',I5)
        NP=3
        XI(1)=B
        FI(1)=FLOR(XI(1))
        XI(2)=(B+E)/2
        FI(2)=FLOR(XI(2))
        XI(3)=E
        FI(3)=FLOR(XI(3))
        D(1)=-1
        D(3)=-1
        IM=2
90      IB=MAX0(2,IM-1)
        IE=MIN0(NP-1,IM+3)
        DO 100 I=IB,IE
        D(I)=ABS(FI(I)-FI(I-1)-((XI(I)-XI(I-1))/(XI(I+1)-XI(I-1)
     #))*(FI(I+1)-FI(I-1)))
        D(I)=ABS(D(I)*(XI(I+1)-XI(I-1)))
100      CONTINUE
C *** FINDING THE NEW IM
        IM=2
        DM=D(2)
```

```
        DO 120 I=3,NP
        IF(DM.GT.D(I))GOTO 120
        DM=D(I)
        IM=I
120     CONTINUE
102     FORMAT(' IN BLI DM, IM',E20.6,I5)
        FSAVE=FI(IM)
        XSAVE=XI(IM)
        XP=.5*(XI(IM)+XI(IM+1))
        XM=.5*(XI(IM)+XI(IM-1))
C *** SHOVE THE STACK UP
        NMOVE=NP-IM
        J=NP
        DO 160 I=1,NMOVE
        XI(J+2)=XI(J)
        FI(J+2)=FI(J)
        D(J+2)=D(J)
160      J=NP-I
        XI(IM)=XM
        FI(IM)=FLOR(XM)
        XI(IM+1)=XSAVE
        FI(IM+1)=FSAVE
        XI(IM+2)=XP
        FI(IM+2)=FLOR(XP)
        NP=NP+2
        IF(NP.LT.N-2.AND.NP.LT.198)GOTO 90
        IF(NW.NE.1)RETURN
        WRITE(*,1985)NP
1985    FORMAT(I5)
        WRITE(*,1986)
1986      FORMAT(18X,'XI',18X,'FI')
        DO 170 I=1,NP
170     WRITE(*,1987)XI(I),FI(I)
1987  FORMAT(2E20.6)
        RETURN
        END
C*******************************************************************************
        FUNCTION GAUSS(X)
C*******************************************************************************
        GAUSS=0.E0
        X2=X*X
        IF(X2.GT.85.E0)RETURN
        GAUSS=1000.*EXP(-X2)
        RETURN
        END
C*******************************************************************************
        FUNCTION FLOREN(X)
C*******************************************************************************
        FLOREN=1000./(X*X+1.E0)
        RETURN
        END
C*******************************************************************************
        FUNCTION EXALG(X)
C*******************************************************************************
        COMMON/PASS/ETA,XSUP(200),NPSUP
      PRINT*,'IN EXALG'
        EXALG=1000.*(1+ETA)*EXP(-X-(ETA*X/2)**2)
        RETURN
```

```
      END
C*********************************************************************
      FUNCTION ALICOS(ALPHA)
C*********************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON/PASS/ETA,XI(100),F(100),NP
       REAL*4 ALICOS,ALPHA,ETA,XI,F
C *** CALCULATES VM INTEGRAL OF DCOS(ALPHA*X)*F(X)
      AIFUN(EF0,EF1,EF2,EF3,X)=DCOS(X)*(EF3*(3*X*X-6)+EF2*2*X
     # +EF1)+DSIN(X)*(EF3*X*(X*X-6)+EF2*(X*X-2)+EF1*X+EF0)
      ALPA=(1+ETA)*ALPHA
      IF(DABS(ALPA).LT.1.D-7)ALPA=1.D-7
      ALPAD=1/ALPA
      ALPAD2=ALPAD*ALPAD
      ALPAD3=ALPAD2*ALPAD
      ALPAD4=ALPAD3*ALPAD
      ALICOS=0
      NPMM=NP-2
      DO 100 I=2,NPMM
      X1=XI(I-1)
      X2=XI(I)
      X3=XI(I+1)
      X4=XI(I+2)
      F1=F(I-1)
      F1=F1/((X1-X2)*(X1-X3)*(X1-X4))
      F2=F(I)/((X2-X1)*(X2-X3)*(X2-X4))
      F3=F(I+1)/((X3-X1)*(X3-X2)*(X3-X4))
      F4=F(I+2)/((X4-X1)*(X4-X2)*(X4-X3))
      EF0=X2*X3*X4*F1+X1*X3*X4*F2+X1*X2*X4*F3+X1*X2*X3*F4
      EF0=-EF0*ALPAD
      EF1=(X2*X3+X2*X4+X3*X4)*F1+(X1*X3+X1*X4+X3*X4)*F2
     # +(X1*X2+X1*X4+X2*X4)*F3+(X1*X2+X1*X3+X2*X3)*F4
      EF1=EF1*ALPAD2
      EF2=(X2+X3+X4)*F1+(X1+X3+X4)*F2+(X1+X2+X4)*F3+(X1+X2+X3)*F4
      EF2=-EF2*ALPAD3
      EF3=(F1+F2+F3+F4)*ALPAD4
      XUB=XI(I+1)*ALPA
      IF(I.EQ.NPMM)XUB=XI(NP)*ALPA
      XLB=XI(I)*ALPA
      IF(I.EQ.2)XLB=XI(1)*ALPA
      FUB=AIFUN(EF0,EF1,EF2,EF3,XUB)
      FLB=AIFUN(EF0,EF1,EF2,EF3,XLB)
      IF(DABS((FUB-FLB)/FLB).LT.1.D-5.AND.DABS(XUB-XLB).LT..5D0)GOTO90
      ALICOS=ALICOS+FUB-FLB
      GOTO 100
90     XA=.5*(XUB+XLB)
       ALICOS=ALICOS+DCOS(XA)*(EF0+XA*(EF1+XA*(EF2+XA*EF3)))*(XUB-XLB)
100    CONTINUE
       RETURN
       END
C*********************************************************************
       SUBROUTINE BREAD(II,NA,EXTWT)
C*********************************************************************
      COMMON/DATA/XI(4096),F(4096),WX(4096),NP
     INTEGER*1 WORDBYTE(1),B1,B2,B3,B4
      DIMENSION IIDAT(8192)
      EQUIVALENCE (IIDAT(1),F(1))
     EQUIVALENCE (WORDBYTE(1),IIDAT(1))
```

```
        CHARACTER*4 II(40),EXTWT
        CHARACTER*64 NA
C *** F(N) IS THE SPECTRUM CURRENTLY BEING FITTED.
      GOTO 15
13      WRITE(*,109)
109     FORMAT(' ENTER THE NAME OF THE DATA FILE')
        READ(*,1943)NA
15      II(1)=NA(:4)
        II(2)=NA(5:8)
        IT=INDEX(NA,'.')
      IF(NA(IT+1:IT+4).EQ.'WDAT')GOTO 3000
1943    FORMAT(A)
        WRITE(*,1001)
1001    FORMAT(' ENTER BEG CHANNEL, END CHANNEL!')
        READ(*,*)N1,N2
        WRITE(*,1178)N1,N2
1178    FORMAT(' IN BREAD N1,N2',2I5)
        IF(NA(IT+1:IT+3).EQ.'UF')GOTO 1100
       PRINT*,' TESTING FOR SALLY ',NA(IT+1:IT+5)
       IF(NA(IT+1 :IT+5).EQ.'SALLY')GOTO 2000
        IRF=0
        IF(NA(IT+1:).EQ.'LEO')IRF=1
        WRITE(*,*)IRF
        IF(IRF.EQ.0)OPEN(8,FILE=NA,STATUS='OLD',ERR=13)
        IF(IRF.EQ.1)OPEN(8,FILE=NA,STATUS='OLD',
     # FORM='UNFORMATTED',ERR=13)
        IF(IRF.NE.1)GOTO 420
        NREC=(N2-1)/128+1
        NU=128*NREC
        DO 416 III=1,NU,128
416     READ(8)(IIDAT(I),I=III,III+127)
C *** PERMUTING THE WORDS FROM VAX TO MAC STORAGE
      N4=4*N2
      DO 418 I = 1,N4,4
          B1= WORDBYTE(I)
          B2 = WORDBYTE(I+1)
          B3 = WORDBYTE(I+2)
          B4 = WORDBYTE(I+3)
          WORDBYTE(I)   = B4
          WORDBYTE(I+1) = B3
          WORDBYTE(I+2) = B2
          WORDBYTE(I+3) = B1
418   CONTINUE
C *** END OF WORD PERMUTING
      PRINT'(8I10)',(IIDAT(I),I=2750,2850)
      READ(*,*)ITEST
        N=N2-N1+1
        XE=N2-N
        DO 400 I=1,N
        XI(I)=I+XE
        F(I)=IIDAT(I+N1-1)
400     CONTINUE
      PRINT'(10F8.0)',(F(I),I=1,N)
      READ(*,*)ITEST
        NP=N
        CLOSE(8)
        RETURN
420     READ(8,101)II
```

```
101     FORMAT(10A4)
      IF(II(40).EQ.'FREE')GOTO 2200
        IF(II(40).NE.'HDAT'.AND.II(40).NE.'IDAT'.AND.II(40).NE.'Z4DA')
     # GOTO 195
        IF(II(40).NE.'Z4DA')READ(8,*)NXB,NEND
        NMULT=1
        IF(II(40).EQ.'Z4DA')READ(8,*)NXB,NEND,NMULT
        NXB=N1
        NEND=N2
        I10=10
        IF(II(40).EQ.'Z4DA')I10=20
        NXBS=(NXB-1)/I10
        IF(NXBS.EQ.0)GOTO 145
        DO 144 I=1,NXBS
144     READ(8,156)
145     NIR=NEND-I10*NXBS
        NIR=MIN0(NIR,8192)
        IF(II(40).EQ.'HDAT')READ(8,156)(IIDAT(I),I=1,NIR)
        IF(II(40).EQ.'IDAT')READ(8,157)(IIDAT(I),I=1,NIR)
        IF(II(40).EQ.'Z4DA')READ(8,158)(IIDAT(I),I=1,NIR)
156     FORMAT(10Z8)
157     FORMAT(10I8)
158     FORMAT(20Z4)
        CLOSE(8)
        NP=NEND-NXB+1
        NADD=NXB-1-I10*NXBS
        XE=NEND-NP
        DO 190 I=1,NP
        XI(I)=I+XE
190     F(I)=IIDAT(I+NADD)*NMULT
        RETURN
195      N=0
200   N=N+1
        IF(N.GT.8192)GOTO 220
        READ(8,*,END=220)XI(N),F(N)
        GOTO 200
220   NP=N-1
        CLOSE(8)
        RETURN
1100     CONTINUE
         OPEN(8,FILE=NA,STATUS='OLD',FORM='UNFORMATTED')
         READ(8)N,XE
         READ(8)(F(I),I=1,N)
         PRINT*,' IN BREAD N,XE',N,XE
         N1=N1-XE
         NT=N+XE
         NC=MIN0(NT,N2)
         N2=N2-XE
         N1=MAX0(1,N1)
        N=N2-N1+1
        XE=N2-N
        DO 1120 I=1,N
        F(I)=F(I+N1-1)
        XI(I)=I+XE
1120    CONTINUE
         NP=N
         CLOSE(8)
         RETURN
```

```
2000  CONTINUE
         OPEN(8,FILE=NA,STATUS='OLD',FORM='UNFORMATTED')
       PRINT*,' IN SALLY PART OF BREAD'
       READ(8)(IIDAT(I),I=1,128)
       I1=0
      DO 2010 I=1,300
      READ(8,END=2015)(IIDAT(I1+J),J=1,128)
      I1=I1+128
2010  CONTINUE
2015  N=I1+J-1
      PRINT*,' AFTER READING DATA N IS',N
        NXB=N1
        NEND=N2
        NP=NEND-NXB+1
        NADD=NXB-1
        XE=NEND-NP
        DO 2190 I=1,NP
      XI(I)=I+XE
2190    F(I)=IIDAT(I+NADD)

2200    CONTINUE
      DO 2210 I=1,15
        READ(8,'(10A4)')II
2210  WRITE(*,'(1X,10A4)')II

        NMULT=1
        NXB=N1
        NEND=N2
      I10=8
        NXBS=(NXB-1)/I10
        IF(NXBS.EQ.0)GOTO 2245
        DO 2244 I=1,NXBS
2244     READ(8,156)
2245     NIR=NEND-I10*NXBS
        NIR=MIN0(NIR,8192)
        READ(8,*)(IIDAT(I),I=1,NIR)
        CLOSE(8)
        NP=NEND-NXB+1
        NADD=NXB-1-I10*NXBS
        XE=NEND-NP
        DO 2290 I=1,NP
        F(I)=IIDAT(I+NADD)*NMULT
2290     XI(I)=I+XE
      RETURN
3000  CONTINUE
      EXTWT='YES '
      OPEN(8,FILE=NA)
      DO 3010 I=1,4096
      READ(8,*,END=3020)XI(I),F(I),WX(I)
      WX(I)=SQRT(WX(I))
3010  CONTINUE
      NP=4096
3020    NP=I-1
      RETURN
      END
      INCLUDE SMSQ.FO
```

Appendix B

ROBFIT users guide

Robust Curve Fitting Code, ROBFIT: User Guide and Performance Enhancement Report

R.L. Coldwell and G. J. Bamford

University of Florida

Institute for Astrophysics and Planetary Exploration

One Progress Boulevard

P.O. Box 33

Alachua, FL 32615

Tel (904) 462-6331

ABSTRACT

This one document provides a User Guide, a presentation of test cases, and an analysis of performance enhancements on the IBM 3090 Vector Facility for ROBFIT. ROBFIT is a package of codes which utilizes splines with optimized knot locations, model peak shapes, and background identification tools to provide robust fitting to complicated data such as encountered in spectral analysis.

Table of contents

1. Introduction

This report is both a user guide and a summary of enhancements and changes involved in migrating the curve fitting code ROBFIT from its development environment of personal computers to the IBM 3090-400/VF environment. The work has been carried out by utilizing an IBM PS/2 model 50 microcomputer and a telephone link to the University of Florida North East Regional Data Center (NERDC) 3090 system. The migration involved three distinct stages of development. The first stage was to create a menu driven code with the option of running either interactively under VM/CMS or in batch mode under MVS, providing the user with all the advantages of the mainframe environment. Secondly, as certain aspects of ROBFIT require graphical output, the next stage was to develop a machine independent graphics interface for the code. We show how this machine independence can be achieved in a very simple fashion without any additional graphics packages. Finally, we investigated the performance enhancements that can be gained by using the vector facility on the 3090 system. We detail this vector speedup and some surprising scalar speedup results that the migration process uncovered.

2. Overview of ROBFIT

The robust fitting code ROBFIT was initially conceived in 1979. Its first use was as a nuclear physics analysis package. Complex nuclear spectra with up to 500, frequently overlapping, peaks in 4000 channels, were analyzed on the NERDC IBM 3033 using overnight runs and batch processing. Free CMS use supplied by NERDC led to the development of SMSQ, the minimizing routine which enables the present code to run efficiently (see refs 1,2). The code then migrated to various PC's for portability and further speedup. Running ROBFIT on a variety of machines has produced a compact and intrinsically efficient code. This early work helped develop a fitting technique that represents the peaks and background of a spectrum with the minimum number of cubic splines. An illustration of the use of the code has been the analysis of data taken on the Antarctic research program to search for supernova gamma rays, ref. 3. Here ROBFIT was run on a Macintosh II and provided a high performance nuclear analysis package.

Although the code was originally for use in the analysis of nuclear spectra, ROBFIT can be used for any general spectral analysis. The wide variety of uses for ROBFIT have motivated us to develop a more usable mainframe version of the code

which, by nature of its development, will be fast in operation. The present version runs on the NERDC IBM 3090 machine and makes use of the vector facility for speed enhancements.

The algorithms used by ROBFIT can be summarized as follows. It is assumed that the data consists of two distinguishable parts called background and foreground. The background consists of slowly varying features (e.g. bremmstrahlung and Compton scattering in nuclear physics spectra), while the foreground consists of rapidly varying features such as photopeaks, escape peaks, and backscatter peaks.

The background is fitted over the entire spectrum as a set of cubic splines with adjustable knots. Fitting over the entire range allows these features to be fitted continuously with fewer constants and more accurate representations than is possible when they are fitted in short sections along with the foreground. The algorithms used are data compression, similar to that described in ref. 4, and second, a minimization algorithm SMSQ, an extension of the Marquardt procedure (also ref. 4, p. 523). This procedure allows efficient optimization of the knots in the exponentials of the splines as well as of all other more linear parameters.

The foreground typically consists of a relatively small number of peak types, that is, a few characteristic shapes which may be repeated many times. These shapes are fitted by a shape-finding routine as a collection so back-to-back cubic splines. These are then, in turn, fitted to the spectrum by first finding the type of peak to try, then fitting the peak type to the data by varying the peak height, location, and width within user-specified limits. Clumps of overlapping peaks are fitted simultaneoulsly. The principle algorithms used are a fast residual locator for finding the peaks, a routine for systematically controling the allowed widths (ref. 2), and SMSQ (mentioned above).

2.1 General operation of ROBFIT

The routines collectively called ROBFIT have been developed to provide a powerful general tool for spectral fitting. A typical spectrum may contain thousands of channels (individual histogram bins) with numerous peaks which vary in both strength and width. In addition these peaks can sit on top of a complex backgournd. ROBFIT has been developed to ease the analysis of such spectra. ROBFIT provides six modes of operation, three for curve fitting and three for display of the data and fits. For fitting, code is provided:

i. for generating a standard peak,
ii. to fit the background alone,
iii. to fit the complete spectrum.

For display, code is provided:

i. to display the raw data,
ii. to display the standard peaks,
iii. to display the full spectral fit.

Figure 1 gives an overview of how ROBFIT is used in the analysis of a spectrum while section 3 provides a more detailed description of the above operation modes. ROBFIT can be run either interactively or in batch mode. Within the body of this report we only discuss the interactive use. Appendix C provides all the details of running under MVS. In that setting, the user must download the output files from the MVS datasets before any viewing of the curve fit can be performed. However, the user also then has the option of printing graphical information on the high quality PL3820 laser printer.

2.2 Interacting with an IBM PC (XT,AT,PS/2)

All ROBFIT routines are supplied on PC compatible diskettes. We assume that the personal computer is running PC-DOS or MS-DOS. The ROBFIT files need to be loaded onto the 3090 before the codes can be run. In order to accomplish this the ROBFIT diskettes contain a communication package NCGRAF BAS which allows the user to access the mainframe from the PC. In order to operate the communication package the user must have the computer RS232 port connected to a modem, and have access to a telephone link to the local IBM mainframe. Utilizing a communication package that operates under BASICA enables ROBFIT to be run on many machines with minimum complexity. On running the BASIC code NCGRAF the user can upload, download, run ROBFIT and perform PC graphics displays by communicating with the IBM mainframe operating CMS. Thus the user can input the codes and data from PC discs, compile the programs, run them and display the results using his or her PC as a dedicated graphics workstation.

To perform the display tasks the PC must have a screen capable of performing graphics operations. The graphics display is primarily for use as a quick look-and-see tool. Running under BASIC means it is limited in its intrinsic resolution. A user

wishing to study more finely detailed plots can run the batch version of ROBFIT which has the option of selecting full PL3820 laser printer resolution (batch operations are treated in Appendix C).

2.3 Getting started

Among the ROBFIT routines supplied on the floppy is a communication program called NCGRAF BAS which runs in BASIC and provides the means for uploading, downloading and running ROBFIT.

NCGRAF is activated by first typing BASICA 'return', then LOAD"NCGRAF 'return', then RUN 'return'. The NCGRAF menu will then be displayed. Before NCGRAF can communicate with the mainframe the modem baud rate must be set. We have fixed this at 1200bps within the code, however, the user can edit NCGRAF.BAS and alter line 160 to the required rate if necessary. The second modem parameter that must be selected can be set within the NCGRAF menu. The method by which the modem transfers data can be either simplex or duplex, this option is set by selecting either 1 or 2 in the NCGRAF menu before entering the modem commands. We use a Packard Bell PB2400PLUS modem which operates in simplex mode, so we select 1 each time we run NCGRAF.

One is now ready to type the modem commands to access the mainframe. To activate the modem a 3 must be entered at the NCGRAF menu stage. After the prompt "Dial the number and press ENTER" is displayed, the user can enter the modem commands to link into the mainframe facility.

Once the mainframe has been accessed and the normal logon sequence finished, the ROBFIT routines can be uploaded. To initiate an upload from the PC to the mainframe the user must perform the following procedures.

While in VM/CMS type: CP TERM PROMPT??

This sets the terminal prompt to ??

Then type: CREATE filename filetype filemode

CMS type; CP TERM PROMPT ??

This sets the terminal prompt to ??.

Then enter SHIFT ~ on the PC which will bring up the communication menu. Select UPLOAD from this menu and enter ?? when asked for the prompt. Then enter the filename to upload. Once the upload write has finished CMS can be entered again by selecting 6 on the communication menu. To end the upload sequence /* must be entered under CMS.

Routines to upload are

ROBFIT	FORTRAN	BKGFIT MENU
GRAPH	FORTRAN	BVRMAI MENU
SMSQC	FORTRAN	GRAF3 MENU
SMSQ	FORTRAN	GRAF4 MENU
VGDPLOT	FORTRAN	GRASHAPE MENU
VPLOTS	FORTRAN	SHAPE MENU
VRMAIN	FORTRAN	VRMAIN MENU
MENURD	FORTRAN	VROBFIT MENU
BVRMAI	FORTRAN	
BMENUR	FORTRAN	NCGRAF BAS
NBCNRO	FORTRAN	Plus all test case files
BNVROB	FORTRAN	beginning with Z.
ROBFIT	EXEC	YT ST
ROBFIT	JOB	COMP ST
CMTOMV	JOB	
CATDS	JOB	
MVTOCM	JOB	
CATLIST	JOB	
CROBFIT	JOB	
UNCATDS	JOB	

Once all routines have been uploaded, all those of filetype FORTRAN must be compiled with the VECTOR option. In addition, the batch routines must be compiled and loaded into a library. This task is accomplished by running the CROBFIT JOB as described in Appendix C.

The ROBFIT EXEC is the file which activates the curve fitting sequence and loads all relevant files. If the user wishes to run without the vector matrix routines SMSQC must be replaced by SMSQ in the load line of the ROBFIT EXEC. Similarly the batch compile must be re-done without the vector libraries and with SMSQC replaced with SMSQ.

On typing ROBFIT the curve fitting codes will be activated and the menu pages can then be operated as described below.

2.4 Using the menu driver

Entry into ROBFIT is performed by running the ROBFIT EXEC under VM/CMS. The starting sequence for the code asks the user to select either an interactive or a batch session. In interactive mode all ROBFIT menus are available while batch mode can be used to run either a background fit a full spectral fit or print the raw data or curve fits. Viewing of data is usually performed interactively, batch mode is being used to produce a high quality output using the PL3820 laser printer. After selecting the session the top-most menu page will be displayed, from this point onward, any of the available operation modes can be selected. In an interactive session, for example, the user can select any one of the six modes of operation. Once the mode has been selected the user has access to that particular mode alone. At this point the mode menu pages can be operated, a number of pages become available each of which is described in full in Appendix A. From the top of the mode menu the user can either select a lower menu page, enabling ROBFIT parameters to be varied, or run/quit the code. The overall hierarchy of the menu is shown in figure 2. Appendix A provides a fuller explanation of how to move around the mode menu pages and gives a description of each of the functions within the menus.

3. Procedures for running ROBFIT

The preceding Section describes the procedures required to load and initialize the routine. Here we show how to go about setting up the menus to perform a fit and cover the various operating modes of ROBFIT. Once all routines have been compiled we can initiate the startup sequence of ROBFIT. The first step in this startup is to run the ROBFIT EXEC. The code then asks the user which session is required, interactive or batch. Entering "INTER" will start an interactive session and all the ROBFIT modes will become available. The alternative response, "BATCH" will initiate a batch session where the user can access the background and full spectral fitting modes and certain laser printer options. Once the session has been chosen the session master menu will be displayed, from this point the user can select any one of the available modes of operation. A mode is accessed by entering the LINE NUMBER, RETURN of the required mode.

All procedures described here relate to running the code under VM/CMS. However, the setting up of the mode menu pages is almost identical in a batch session. The operation of a batch session is outlined in Appendix C. The user must note that all viewing of curve fits, standards or background fits has to be done interactively using

the graphics routines outlined below. Final hard copies can be made on a PL3820 laser printer by running a batch session.

3.1 Raw data graphical display (RAWDD)

This section provides the procedure for viewing the raw input data, that is prior to any ROBFIT curve fitting or standard generation. The display is activated by selecting the RAWDD mode from the session master menu. In addition to being able to view the input data of a single spectrum, the user can opt to view two spectra simultaneously. The user also can opt to plot the difference between the two spectra. This option is useful for comparing data with a small signal in one of the spectra, an example being the detection of the supernova signal illustrated in reference 3. Another use of the raw data display is to select a region of the spectrum for use as a standard. If the spectrum contains peaks of unknown shape the a standard must be created from the data, for this purpose a relatively clean region of the spectrum containing a peak is required. The raw data display allows the user to scan the input data for such a region. Operation of the standard generation is outlined in Section 3.2.

There is a further option in the display which allows the user to reconstruct corrupted data. *If for some reason the data has been affected by, for example,* machine transfer problems the user can elect to smooth the data. Regions of the data can be removed by hand. The code will interpolate across the selected region using user defined data.

The following gives an overview of the procedures that must be followed to set up the display menu for operation. All menu pages are described in full in Appendix A.

Step 1. Setting up the general information (RAWDD1)

On entering the RAWDD mode menu, the user has the option of viewing three sub-menus. In the first instance the general information needs to be modified. On selecting the RAWDD1 sub-menu page by entering 1, the user is ready to proceed with the first step in setting up the display. Line one of the sub-menu requires file names to be entered, it is structured to accept two filenames, each of these files must contain data of the format outlined in Appendix B. The second file is used only if the user wishes to compare two spectra, if there is only a single file to view then a "\" is entered for the second filename.

The next stage is to input the initial and final channels of the display region and the minimum and maximum peak heights if known. If these values are not set then the display will select them from the data. Default options are explained in more detail in Appendix A.

The final stage of setting up the display is to simply select the type of y-axis plot required, either logarithmic or linear, and set a title for the graph. Provided the user does not wish to correct data or plot a difference between two spectra the above is all that is required to initialize the display. The user would now go back up to the RAWDD mode master menu page and run the raw data display by entering 100. After the graph has been displayed the menu options can again be accessed by entering "NO" to the exit prompt which appears at the end of the plotting.

Step 2. Setting up the difference information (RAWDD2)

This step must be followed if the user wishes to study the difference between two sets of input data. The difference plot is activated if a "YES" is input into the first line of the RAWDD2 sub-menu page. If "NO" is entered then the following lines of the menu are ignored by the display driver.

Selecting the DIVIDE BY ERROR option will divide all individual channel residuals by the errors calculated from the sum of the two scaled data channels. This option is for viewing purposes only. It reduces the wide fluctuations of the data that can occur when taking the difference between two spectra.

Next the filename for the sum of the two raw data files is entered. The sum file is used in the full spectral fitting of the difference output as it contains the errors for each of the difference data channels. The file name for the difference data is then entered. This file contains the results of the subtraction of the two raw data files. The user then specifies the minimum and maximum peak heights, if known and selects a logarithmic or linear fit.

Finally the user must specify the scaling parameter for each of the spectra. These values are used to normalize the two spectra. If there is no need to normalize then the default option selects 1 as the scaling for both files. Otherwise the code will scale the channel values of the second file by VALUE1/VALUE2.

Step 3. Setting up the correction information (RAWDD 3)

This third step is performed only if the data need correction. Again if "NO" is entered in line one of the RAWDD3 sub-menu then the following lines of the menu are ignored by the display driver.

The second line contains the range over which the correction is to take place. The third line contains the filename for the corrected data and the fourth line is the starting and ending points for the correction. ROBFIT will then linearly interpolate across the selected region. All corrected data must be stored in files which have a filename specified as 'Filename' UF, these files can then be entered in the ROBFIT full spectral fitting routines.

3.2 Standard Generation (STGEN)

In its peak fitting phase, ROBFIT requires knowledge of the peak shapes it expects to find within the data. These prototype peaks, termed standards, can be of any shape. The standard generating mode allows the user to either create a standard from the raw data or to use one of the three most common shapes as provided with ROBFIT. Each standard is fitted to a polynomial plus a set of back-to-back cubic splines. The polynomial is assumed to represent the background under the standard while the splines represent the standard peak. Using this technique allows complex shaped standards to be generated. STGEN runs interactively under CMS.

Step 1. Setting up the general information (STGEN1)

On entering the STGEN mode menu pages the user must first select the general information page for initialization. The first stage is to select the filenames in which

i) the standard coefficients are to be placed, and
ii) the output in which the graphics data are to be placed.

The standards coefficients file will be read by the full spectral fit routines and used in the peak fitting stage while the graphics file is for input into the standard display mode STDIS.

The next stage is to select the standard type. Three special peak types are provided with ROBFIT, Gaussian, Lorentzian and Voigt (Ref. 5). This last shape is a Gaussian convoluted with a Lorentzian with the ratio of the width of the Lorentzian

to the Gaussian is supplied by the user. If the data contain peaks of such shape as these, the user can generate the appropriate standard by selecting among GAUS, LORE or VOIG. The code will then proceed to generate the selected standard. If, however, the peak shapes within the real data are unknown then a standard can be made from the raw data. To select a fit to the real data, FITD is specified, this option is explained further in *Step 2*. All standards are normalized to have height 1 and full width half max 1, enabling different standards to be fitted simultaneously.

Finally the user must select the number of background coefficients (maximum 4) and splines (usually 3 to 5) to be used in the standard representation. Complex peaks and background will require more coefficients for a faithful reproduction of the shape. If the user is selecting a standard from the real data it may be that a number of iterations of fitting first a standard then the data will be required before a satisfactory standard can be found. Studying how the code fits a particular standard to the data shows how well STGEN has represented the background under the selected peak.

Step 2. Entering the fit type information (STGEN2)

This next step sets up some initialization parameters for the Voig t and data fit options of the standard generator. These only need to be set if the corresponding options have been selected in step 1, otherwise they are ignored by the standard generator. The first parameter to be set is the Voigt η (ETA) defined by $\eta = \Gamma_G/\Gamma_L$ where Γ_G is the width of the Gaussian of unit height that is convoluted with a Lorentzian also of unit height and width Γ_L.

The second stage is to set the raw data pointers. If the peak shapes within the raw data are unknown then a standard must be created from the real data. In order to do this the user must select a relatively clean peak within the data. That is a peak which is well separated from any other peaks or complicated background regions. This can be done with the raw data display mode. This peak will then be fitted using a polynomial for the underlying background and a set of back-to-back cubic splines to represent the peak. Coefficients from this fit are then used in the full spectral fitting routine to determine if peaks of this shape reside within the data. In order to fit to the real data the standard generator needs only to know the raw data filename and the channel range over which it has to fit the standard.

3.3 Displaying the standard (STDIS)

Once the standard has been created it can be viewed using the standard display option STDIS. With this display routine the user has the option of viewing the curve fit, the underlying background and the splines which make up the standard. This option must be run interactively under VM/CMS.

Step 1. Setting up the general information (STDIS1)

The first step is to enter the general information sub-menu STDIS1 and specify the name of the file containing the graphics output of the standard generator. The beginning and ending channel numbers and desired minimum/maximum y-scale range can then be entered. These parameters will default to the ranges within the data if they are not specified. Appendix A shows how to set up these default options. Finally the user must specify either a linear or logarithmic y-scale for the fit.

Step 2. Setting up the display options (STDIS2)

In this step the various plotting options are set. The user specifies either a "YES" or "NO" to select the viewing options.

3.4 Background fitting (BKGFIT).

Certain spectra are best analyzed by first determining the underlying background function. By calculating the background shape the spectral fitting of the data will be enhanced and in some cases a more accurate fit will result. This is especially so when complex backgrounds are involved where confusion between peaks and background can cause problems. BKGFIT can be run either interactively or in batch mode.

Step 1. Setting up the input parameters (BKGFIT1)

The first step in initializing the background fitter is to set the input parameters of sub-menu BKGFIT1. Here the user must first select the number of background-constants that are to be used in the fitting. This number will depend on the complexity of the background. The method recommended to determine the optimal number of constants is to perform a set of background-fits to the data, each run containing a few more constants. Once the optimal number has been reached the

values of any additional constants will be small in comparison to the rest. By monitoring these numbers together with viewing the fitted background the user can determine the background shape to a level that will be satisfactory for input into the full spectral fitting code.

The next stage in setting up the input is to specify the CUTOFF parameter. This number is used to determine the degree of robustness in the fit. The fitting is ended when the largest 5 point sum, scanned over all the data, differs from the fit by CUTOFF*σ where σ is the standard deviation of the five points.

Now the filename of the raw data can be input together with the channel range over which the fit is to be performed. If there has already been some processing of the data and a file containing background-constants has been created, then these data can be used as a starting point for the new fit. All the user has to do is specify the filename containing the background-constants from the previous fit. If there is no background-constants file, as in the first iteration of fitting, "NONE" is input for the filename. In running the fit the data can be fitted on a logarithmic scale. This option being set by specifying "YES" on the line containing the background-constants filename. A "NO" will perform a linear fit.

The user next selects the weight to place on the data points. Selecting a 0 will take the error in a channel equal to $\sqrt{\text{data value.}}$ If a 1 is selected then an averaging calculation is performed for the whole spectra and used to calculate the error at a given point. Selecting 2 allows the user to read the error values in from an external file. The filename containing the error values is then entered on the next line. This is only necessary if option 2 has been selected for the weights. The file format is given in Appendix B.

Step 2. Checking the output parameters (BFGFIT2)

In this step there is no input required from the user. Here the output files in which the background-constants and fit information need to be checked. The first of the two files is the output file for the background-constants and is called BKGCONS CN. This file can be edited and/or renamed to any user specified name. If the file is to be used in further fitting of either the background or peaks, then this new filename must be input into these codes.

The second file called BKGCONS GR contains the graphical output required to display the standard when running FSPDIS.

3.5 Full Spectral Fitting (FSPFIT).

These routines make up the core of the fitting procedure, here a complete analysis of the data is performed. After cycling through this code the peak positions, widths and strengths are fully determined and the background shape will have been calculated. The code is able to start with or without information on the background and/or peak parameters. If, for example, the background has been determined as described in the previous section then this information can be entered into the full spectral fit at the beginning of the run. In doing so the code will not have to redetermine the background coefficients and the spectral fit will be speeded up. Alternatively the positions and/or widths of certain peaks may be known beforehand. In such a case this information can also be linked into the program in a similar fashion, again resulting in operation speedup and possibly increasing the accuracy of the final peak parameters.

The code uses the standard peak shapes to search for additional peaks within the data. Each individual peak is determined by minimizing a weighted sum of differences between the data and the background with respect to the peak height, location and width along with up to nine neighboring peaks already present. After enough peaks have been found the weights and background are redetermined. Then the old peaks are refitted and new peaks are added and the cycle repeated until the largest 5 point sum is less than a user specified standard deviation. This iterative procedure is outlined in figure 3.

This section provides a brief outline of how to go about setting up a particular problem. Some of the steps in the following may be redundant if information on the data is known a priori. For example, peak widths may be known to a high accuracy or their variation with respect to channel number may be known. If this is the case then such information can be incorporated into the spectral fit at the outset. FSPFIT can be run either interactively or in batch mode, the operation of the output files in a batch session is outlined in more detail in Appendix C.

Step 1. Setting up general fit information (FSPFIT1)

The first step is to define the maximum number of peaks that are to be found within the spectrum, on finding this number of peaks the code will stop.

The second stage is to define the range of cutoff values over which the fit will operate. Three values are specified on line two of the menu page. The first value being the starting cutoff, the second being the ending cutoff and the third being the

number of steps to take between the two extremes. For example 2,1,2 will run the fit down to a cutoff of 2 in its first iteration, then 1.5 in its second,and finally end at a cutoff of 1.

Next is the definition of how the routine is to handle the width variation and background fitting. Two parameters are defined on line three of the menu page. The first value specifies whether the fit is to proceed with variable width peaks, the variation being taken over the limits set on menu page FSPFIT5. This type of fit is initiated by specifying VWID. Alternatively a FIXD option can be specified where the widths are held constant at the values determined from menu page FSPFIT5.

The second parameter sets up the background fitting operation. Four choices are available: CONT, NONK, NOBF and FIXK. CONT tells ROBFIT that on each background re-fit an additional knot is to be added to the background. NONK sets up a mode of operation where no new knots are to be added to the background. NOBF selects the no background fit mode of operation. The FIXK option allows the user to hold the knots stationary. This is useful for comparing data with the same underlying background structure. In this case, knot positions can be held constant while heights are allowed to vary, to account for differing normalizations. With this option comes the ability to fit certain backgrounds with fewer constants, thus increasing the speed of operation while maintaining the accuracy.

The next stage is to limit the number of background constants allowed for this particular fit. These values are entered on line four of the menu. For a complex background a large number of constants may be required, the code allows up to 100 constants to be used. The user selects the minimum and maximum number of constants. No peaks will be added until the background has been fitted with this minimum number of constants. Following this initial background fitting the code will cycle, succesively fitting peaks and adding background constants, until the maximum number has been reached.

The user then selects the weight to place on the data points. Selecting a "0" will take the error in a channel equal to $\sqrt{\text{data value}}$. If a "1" is selected then an averaging calculation is performed for the whole spectra and used to calculate the error at a given point. Selecting "2" reads the error values from an external file. The next line contains the filename of the file containing these error values, with the format of this file as shown in Appendix B.

Step 2. Setting up the input data (FSPFIT2)

When starting off a particular fit the first thing to do is to set up the pointers to the information that will be input into the fit. On line one the user enters the name of the file containing the raw data. The format with which ROBFIT expects to find this data is given in Appendix B.

Once the data file has been specified the user then has to select the channel range over which the fit will be performed. This range is entered on line two of the menu. The fit need not start at channel 1, any offset will automatically be taken care of.

The next stage is to specify whether there is a peak file that can be used in the fit. This would be the case after a number of cycles of the program where certain peaks have been accurately determined. The filename containing these values is entered on line three of the menu. Again the file must be specified as 'filename' PK. Here PK specifies a peak input file. If there is no peak file then the users enters "NONE" for the filename. The format of the peak file is outlined in Appendix B. Position,width and strength, and the corresponding errors of any peak can each be edited by hand before input into ROBFIT. In the process of operation ROBFIT will create a new peak file. This new file will contain the additional peaks found during the fit. It can be arranged so that the new peak file overwrites the input peak file, a procedure which will be outlined later in this section. A qualifier must be placed after the filename. If the input errors on peak positions, widths and strengths are known accurately then a "YES" is selected if not then "NO". If the qualifier is NO then ROBFIT will recalculate these parameters from other information entered in the fit.

The next stage in setting up the input files is to specify the file containing the background-constants. As above, if there is no input file then "NONE" is entered for the filename, otherwise the format is 'filename' CN. Also as above, these are constants that have been calculated on a previous run of the code. If no file is specified then ROBFIT will recalculate the background from scratch. Depending on the complexity of the background this could add considerably to the execution time. In its operation ROBFIT will create a new constants file and again it can be arranged so that the input constants file is overwritten as the program performs the fit. There is also a qualifier that goes with this line, however this time it specifies how the program is to fit the background. If a "YES" is selected then ROBFIT will perform a logarithmic fit to the background if "NO" then a linear fit will be performed.

Step 3. Setting up the output files (FSPFIT3)

The first stage in setting up the output files is to define the file in which the background-constants are to be placed. This file must have the name 'filename' CN. It may be that the user wishes to overwrite the old input constants with constants from the new fit in which case the filename is set to the input filename. In doing this ROBFIT will initially read in the input constants and start the new fit with these values. Then on its output, after it has cycled through one complete fit, it will output the new constants to the old filename.

The next stage is to define the output peak file. After performing a fit the program writes out the peak positions, widths, strengths and their corresponding errors into the file that is specified on line two of the menu page. Again there is a file naming convention: the name must have the structure 'filename' PK. As with the background constants, the program can be set up to iterate on a peak file and again the output file can be named as the input file.

Here the user must define the graphics output file. Again there is a fixed naming convention, the file must be named 'filename' GR. The graphics file contains the information used in the display of the full spectral fit, it is this file which is specified in line one of menu page FSPDIS1.

Step 4. Setting up the standards (FSPFIT4)

With every fit there is a requirement that at least one standard be defined (without a standard ROBFIT can not function). The name of the file containing this first standard is defined in line one of the menu page. Standards are generated using the standard generation mode of ROBFIT, see the STGEN menu pages. Again these standard files have a naming convention: all files are of the form 'filename' ST.

If the spectrum being fitted has within it a number of different peak shapes, each shape must have a standard defined. For example in nuclear data, the photoelectric peaks are of one shape while the Compton background is of a different shape, hence each will require a standard to perform a correct fit. The files containing these additional standards are added to the menu page.

Once the standards have been defined the next stage is to detail how the widths of these different shape peaks are to vary over the channel range being fitted. This information is input into menu page FSPFIT5. Two lines of the menu page are required for each standard. The first line defines the starting values for this width

variation the second line defines the ending values. See Appendix A for further details. Once this has been set up the standard definition is complete.

3.6 Display of the curve fit/background fit (FSPDIS).

This mode of operation gives the user the ability to view a full spectral fit or a background fit. The structure of these menu pages is identical to that of the standard display option described above. Here the file containing the graphical output from FSPFIT or BKGFIT is input at *Step 1* of that procedure.

FSPDIS runs both interactively under VM/CMS and in batch mode under MVS. The batch option can be used to produce a high quality laser printer hardcopy. The graphical information from an interactive session can be viewed using NCGRAF.

4 Test cases, demonstration of the algorithm

This section describes the tests that have been performed on the spectral fitting routines. It is hoped that these tests will serve as a guide for the user in his or her analysis. The tests show the accuracy that can be achieved under certain circumstances and detail an analysis methodology that can be used on real data.

The four major areas that the testing was designed to probe are,

i). Reproducibility of a single generated peak.
ii). Peak detection level with respect to noise.
iii). Detection of peak separability.
iv). Complex background problems.

For each of these we have provided test cases which the user can run. Appendix D contains the details required to set up and run these test cases.

4.1 Single peak detection efficiency

In order to make these tests more realistic, generated peaks have been superimposed upon an exponentially decaying background. The first test was to generate a large peak in two separate regions of this background to show that under ideal circumstances the routines do accurately reproduce the underlying structure. The x-axis channel range runs from 1 up to 1000 channels and peaks are generated at two positions along the axis, one at channel 150 and the second at channel 850. The generated peaks have a width of 46.9 channels and a strength (total number of counts under the peak) of 10000 (which corresponds to a peak height of 200 channels). Both have Gaussian profiles. Results of ten

separate runs for each position are shown in Table 1. As can be seen at this level the routine does indeed reproduce the generated peak. A test case of the above single peak fitting is detailed in Appendix D.

Peak Position	Width (channels)	Strength (counts)
150.14 ± 0.62	46.40 ± 2.57	9357 ± 1173
149.76 ± 0.46	51.00 ± 1.13	11057 ± 226
149.06 ± 0.89	46.06 ± 3.01	9450 ± 1448
151.18 ± 0.61	43.04 ± 2.10	7439 ± 653
151.71 ± 0.60	44.76 ± 2.09	8258 ± 712
150.34 ± 0.70	47.07 ± 2.70	9500 ±1256
150.69 ± 0.59	47.34 ± 2.44	9904 ± 1136
150.61 ± 0.55	48.54 ± 1.79	9920 ± 632
152.81 ± 0.65	48.27 ± 1.83	8376 ± 442
150.61 ± 0.74	49.10 ± 2.62	10211 ± 1263
850.00 ± 0.22	46.67 ± 0.42	10018 ± 108
849.63 ± 0.22	46.96 ± 0.43	9997 ± 108
849.74 ± 0.22	47.15 ± 0.43	9947 ± 108
850.03 ± 0.22	46.66 ± 0.42	9926 ± 107
850.06 ± 0.22	47.30 ± 0.43	9825 ± 107
850.22 ± 0.22	47.48 ± 0.43	10067 ± 108
849.99 ± 0.22	47.65 ± 0.43	10062 ± 108
850.21 ± 0.22	46.50 ± 0.43	9986 ± 108
849.92 ± 0.22	46.83 ± 0.43	9974 ± 108
849.80 ± 0.22	47.05 ± 0.43	9903 ± 107

Table 1. Resutlts of ten runs at two extreme regions of background. Peaks were generated at 150 (upper half of table) and 850 channels (lower half) with a width of 46.9 and a strength of 10000.

The next stage of testing involved the reduction of the strength of the above peaks just discussed to monitor the routines' ability to identify small peaks.

An important factor in the ability to find a peak is the size of the peak with respect to the background fluctuations. The background has been generated as an exponential function with a corresponding error at a given channel of

$\sqrt{\text{number of counts in channel.}}$ At the 150 position, the average level of the background is approx. 240 $\pm$ 15 counts while at the 850 position it is approx. 7 $\pm$ 3 counts. These represent two extreme regions of background and thus provide a realistic test for the routine. At each position a peak was generated with various strengths until the routine could no longer unambiguously detect a single peak. In general, peaks are added until the ratio of their strength to the standard deviation in the counts under the peak is less than a cutoff value set by the user. In this test the cutoff was set at 4, which means that a chance, positive fluctuation would be considered a peak only if it were larger than four standard deviations. This cutoff value is difficult to determine in real data. However, a technique of first determining large peaks within a spectrum with a large cutoff then working down to smaller peaks with successively smaller cutoff values works well in practice. Having determined this cutoff level, the peak strength variation was carried out with this value held constant.

At the 150 channel position peaks down to a strength of 1150 are detected, which corresponds to a peak height of 23 $\pm$ 5 channels. This is approx. 1.5 times the size of the background noise level. At this level roughly 80% of peaks could be found; the results of five separate runs are detailed in Table 2. As can be seen the code was able to reproduce the peak structure fairly well even at this level. At the 850 position, peaks could be distinguished unambiguously down to a strength of 300 or a peak height of 6 $\pm$ 2 channels again the order of 2 times the noise level.

Peak Position (channels)	Width (channels)	Strength (counts)
149.11 $\pm$ 3.49	39.96 $\pm$ 9.67	905 $\pm$ 259
148.35 $\pm$ 3.48	47.66 $\pm$ 9.19	1168 $\pm$ 255
146.30 $\pm$ 2.44	38.58 $\pm$ 6.34	1199 $\pm$ 220
153.99 $\pm$ 3.22	30.00 $\pm$ 8.36	606 $\pm$ 186
No peak found		
152.65 $\pm$ 2.92	42.50 $\pm$ 7.75	1155 $\pm$ 240

Table 2. Results of the five generated peaks with position = 150, width = 46.9 channels, and strength = 1150. In each run the peak is generated on a random, expoentially decaying background.

Test cases for the preceeding minimum detectable single peak fitting are

detailed in Appendix D.

In order to study the effect of peak width on detectability a single peak of width 6 channels was generated at the 850 position. The fitting code was able to unambiguously select out a single peak when the strength was set to 70 which corresponds to a height of 11 ± 3 or approx. 4 times the noise level. The results of these runs are shown in Table 3. Depending on the peak width and the background level the routine can determine unambiguously peak positions and widths for peaks of height between 1.5 and 4 times the noise level. A test case is described in Appendix D.

Peak Position (channels)	Width (channels)	Strength (counts)
849.51 ± 0.58	7.17 ± 1.39	89 ± 18
849.72 ± 0.63	5.83 ± 1.55	61 ± 17
849.11 ± 0.64	5.09 ± 1.57	58 ± 19
851.16 ± 1.26	7.52 ± 3.17	40 ± 18
850.41 ± 0.59	5.47 ± 1.43	58 ± 16

Table 3. Results of five runs with a peak at position = 850, width = 6 *channels, and strength* = 70 *counts.*

The limits quoted above are conservative in that they require a very low level of spurious peak detection. If, however, we allow a certain number of spurious peaks to appear then we can run the cutoff value down to a lower level, thus enabling the fitting routine to pick up smaller peaks. With real data this is not uncommon in that the region of interest is usually narrow in channel width and the chance of spurious peak interfering is low. This of course depends entirely on the type of data being analyzed. For the present analysis the cutoff value could be taken down to a value of 1. At this level a spurious peak was found genearlly in every 120 channels, a frequency which we deemed acceptable. The strength of the peak was then reduced resulting in a new detection point for the peaks of width 6 channels. Now peaks down to a strength of 25 or peak height 4 ± 2 channels, roughly 1.3 times the noise level could be detected. These results can be seen in Table 4. Again the code faithfully reproduces the generated peak shape even at this low level. A test case of this narrow-width, single-peak fitting is detailed in Appendix D.

Peak Position (channels)	Width (channels)	Strength (counts)
848.29 ± 1.44	9.81 ± 3.42	49 ± 19
848.46 ± 2.30	7.14 ± 5.83	20 ± 18
848.27 ± 1.91	4.93 ± 1.17	16 ± 13
No peak found	----------	------
851.32 ± 1.18	3.64 ± 3.01	14 ± 12

Table 4. Results of five runs with a peak at position = 850, width = 6 channels, and strength = 25 counts.

Another important input to the spectral fit is the number of background constants one allows in the fit. The results above were taken with the minimum number of background constants, 4. As more constants are added the peak detectability will be degraded somewhat as the background tries to follow the data. Again this choice is highly dependent on the data being analyzed. If the background is complex then more constants will be required, resulting in a compromise with peak detectability.

4.2 Multiple peak detection efficiency

The second topic addressed in our testing program was the level at which the code could detect multiple peaks. Here we concentrated on generating peaks separated by a known amount. From the preceeding results we generated two peaks close to the 850 channel position, each of strength 70. At this strength the program would have no problems in distinguishing the peaks independently at a cutoff of 4. The separation of the peaks was then varied until the program could just discern the two peaks. The peak widths were again set to 6 channels. The results of the limiting case are shown in Table 5. Again the program correctly finds the peak shapes. A separation of 15 channels, roughly 2.5 times the peak width, was required before the two peaks could be unambiguously determined. A test case of the above double peak fitting is detailed in Appendix D.

Peak Position (channels)	Width (channels)	Strength (counts)
849.64 ± 0.51	4.05 ± 1.23	43 ± 17
865.18 ± 0.41	6.25 ± 1.14	84 ± 17
849.60 ± 0.60	5.40 ± 1.10	58 ± 15
No peak found	-----	----
849.13 ± 0.65	5.11 ± 1.24	56 ± 17
864.86 ± 0.43	4.03 ± 1.06	54 ± 15
851.66 ± 0.89	5.49 ± 1.03	35 ± 12
863.93 ± 0.42	3.65 ± 0.98	45 ± 12
850.43 ± 0.57	5.27 ± 1.05	55 ± 14
865.37 ± 0.71	5.80 ± 1.04	49 ± 14

Table 5. Results of five separate runs to study the detectability of two peaks positioned at 850 and 865 (channel number) both with widths = 6 channels and strengths = 70 counts.

As with the previous section this level of separation is a conservative one and smaller separations can be picked up by running the cutoff level to a smaller value. With the cutoff set at 1.5 the routine could distinguish the above peaks when separated by 7 channels or approx. 1.2 times the width. Here however, the user has to contend with the spurious peaks that are picked up at this low cutoff value. Table 6 shows the results of this analysis. A test case of the above minimum detectable separation for double peak fitting is detailed in Appendix D.

Peak Postition (channels)	Width (channels)	Strength (counts)
850.46 ± 0.61	7.00 ± 1.27	102 ± 19
857.98 ± 0.46	3.97 ± 1.03	57 ± 15
850.45 ± 0.73	6.02 ± 1.24	77 ± 18
857.17 ± 0.55	3.67 ± 1.26	44 ± 16
849.13 ± 0.74	5.32 ± 1.21	58 ± 17
856.38 ± 0.86	5.47 ± 1.18	52 ± 16
847.76 ± 2.22	5.10 ± 1.05	16 ± 12
855.53 ± 0.63	6.66 ± 1.28	90 ± 19
851.25 ± 0.81	5.00 ± 1.24	70 ± 19
857.79 ± 0.96	5.00 ± 1.24	46 ± 17

Table 6. Results of five separate runs to study the detectability of two peaks positioned at 850 and 857 (channel number) both with widths = 6 channels and strengths = 70 counts.

As in the previous section the strength of the peaks can be reduced to see how this affects the separability detection. At the cutoff of 1.5 level, peaks of strength 25 or peak height 4 can be detected provided they are separated by 2.5 times the width of the peaks. Thus, there is again a compromise between the separation detection and the strength of the peak with respect to the noise. A test case of this minimum detectable double-peak fitting is detailed in Appendix D. The preceeding tests have been carried out with three and four peaks separated by equal amounts. No deviations from the results have been found in these cases.

4.3 Complex background fitting

The final test case considered is one which involves dealing with a complicated background. The function was an yttrium spectrum chosen for its complexity of structure. In order to fit this spectrum a large number of background constants will be needed. If the full spectral curve fitting code, FSPFIT, is run on data containing such a background, confusion between background and peaks will occur resulting with spurious peaks being among the results. Taking out as much of the background as possible by first fitting it alone will therefore enhance the peak fitting. The level to which this compromise between background and peaks can be taken is dependent on the spectrum under analysis. The user must determine this level by deciding what information is needed from the curve fitting. Using a large number of constants to fit the background will degrade the detection of small peaks within the spectrum while using only a few background constants may leave some background contribution in the peak fitting. In this example we show how a large number of background coefficients can be fitted to the yttrium spectrum.

First the background fitting mode, BKGFIT, is entered and all relevant input parameters are set as outlined in Chapter 3. With the number of background constants set to 10 for the first run, once the background fitter has added in all 10 constants it will exit the fitting. The code would also exit if the fit actually met the cutoff criteria. To eliminate this exit route, we set the cutoff to 20, a low value for this data. After new constants have been added and the fit re-adjusted to take into account these new constants the code will cycle and add another set if constants. This process will be repeated until all ten constants have been added, whereupon then the fitter will exit and write the constants to the BKGCONS file. The magnitudes of all 10 constants are then noted by editing the BKGCONS CN file. The next stage is to re-run the background fitter and add in a further 10 constants by raising the number of fitted constants to 20. Now the old BKGCONS file can be read in at the outset of fitting.

This speeds up the fitting process as the code does not have to re-calculate the first ten constants. Once the 20 constants have been added and fitting stops, the BKGCONS file is again examined and the constants noted. For general purposes the process of adding more constants is then repeated until the following criteria are reached.

At the outset the constants will be relatively large. However, as more and more are added, the magnitude of the higher constants will slowly drop off. A sign that the optimum number of constants has been reached is a rapid drop in size of the added constant. A second indication is oscillation of the size of the constants from one to the next. At this point adding more constants will not improve the fitting. Here we detail only the background fitting up to the point at which 98 constants have been added.

A test case of the above complex background fitting is detailed in Appendix D. As can be seen by running the plotting routines for this fit, more constants would be needed to determine the shape fully. Usually such a spectrum would be fitted with a combination of peak fitting and background fitting. As the object of this test case was to illustrate the power of the background fitting under extreme circumstances, peak fitting was omitted.

5. Performance enhancements on the IBM 3090VF

The performance of ROBFIT has been monitored in both scalar and vector modes on the NERDC IBM 3090 VF. By incorporating into the program the matrix routines from the Extended Scientific Subroutine Library (ESSL), an increase in performance can be achieved for applications requiring relatively large vector lengths. There are three distinct stages to the performance measurements, and improvements. The first is determination of the baseline operation, the second is the study of the effects of a straightforward (i.e. unmodified code) use of the vector compiler and the third is optimization of the code to eliminate or reduce "hot spots" within the program. A note of warning is required when switching between scalar and vector operation. The two modes do not have identical resolution in the minimization routine SMSQ(C). Although both modes will produce comparable results, the operation time can be vastly different. In the following discussion we have normalized the timing information to the number of calls to the matrix inversion routines.

5.1 Initial timing and scalar speedup

In order to determine the relative time spent in each part of the code we have used the interactive DEBUG facility. With the DEBUG timing option switched on it is possible to break down the percentage of time taken up by individual routines.

The results of a timing run are shown in figure 4. Here the fitting routine has been run with a single peak on an exponential background with only four coefficients selected for the background fit. Timing output from the DEBUG program is not shown at this point as such data are dependent upon the DEBUG's action and therefore do not provide a true representation of operation speedup. As can be seen from Figure 4, the majority of the processing time in the original code is being spent in one routine (SMINV). This finding gave credibility to the assumption that enhancements in speed could be achieved relatively simply.

Before discussing vectorization we first detail the scalar enhancements that have been made on the code. Odd though it may seem, the first enhancement involved the specification of no underflows. There are two methods used to avoid listing these errors when running. One is to specify a call to ERRSET which can be used to switch off the error printout. Here however the calls to the error recovery routines still contribute to the timing. Secondly, the NOXUFLOW option can be set in the EXEC which runs the job. The method used has a dramatic effect on operation time for our code. Table 7 shows some timings for a fit to an exponential background

together with a single peak. Here a factor of two in execution time can be gained by specifying the correct option.

Alteration	Before (sec)	After (sec)
Underflow handling	69.0	33.5
Optimization level	33.5	11.1

Table 7. Test timings (IBM 3090-400) for a fit to an exponential background containing a single peak, with the background fit to 30 coefficients. Underflow handling "Before" is with ERRSET, while "After" is NOXUFLOW. Optimization "Before" is Opt level 1, while "After" is Opt level 3.

The second enhancement involved utilization of the OPT 3 level of the VS FORTRAN 2.2 compiler. Table 7 also shows the effects of selecting the baseline option OPT 1 and the fully optimized version of the code compiled under OPT 3. A dramatic gain in speed can be achieved by using the higher OPT level, a factor of 3 or more in our case. For our code such timings are dependent on the number of background constants being fitted. In Table 8 we show the effect of selecting the correct options as the fitting requires more and more constants.

Number of background constants	Before (sec)	After (sec)
10	3.7	1.5
20	19.8	5.6
30	69.0	11.1
40	222.8	28.4
54	527.1	50.1

Table 8. Illustration of the performance gain by use of efficient underflow handling and optimization level (together) as a function of the number of fitted background constants. "Before" is underflows by ERRSET and Opt level 1, while "After" is underflows by NOXUFLOW and Opt level 3.

From these results we see that operation speed of the baseline code depends upon selecting correct error recovery and optimization levels. A large factor in operation speed can be lost by specifying the wrong startup levels.

5.2 Vectorization speedup

The vectorization stage involved two stages. First is simple or "naive" vectorization, achieved by simply passing the code through the vector compiler. Second is replacement of the matrix operations within the code with their corresponding ESSL partners.

For a full test of naive vectorization, we used as the test case the yttrium backgound sample examined in the complex background test case. The code was tested using different numbers of background constants and thus different vector lengths. We timed the ability of the code to add two new constants to the background at vector lengths of 20, 30, 40, 50, 60 and 70. The results of the pass thrugh the vector compiler are shown in Table 9. At this stage ther was no major speed enhancement seen by utilizing the vector compiler. The code in fact was slowed somewhat for vector lengths below fifty.

Number of background coefficients	Scalar (sec)	Vector "naive" (sec)	Scalar with ESSL (sec)	Vector with ESSL (sec)
20	20.1	27.0	14.7	16.4
30	52.5	63.1	35.1	39.1
40	97.2	107.1	52.7	56.2
50	171.1	172.0	62.1	56.2
60	256.3	241.7	77.5	73.8
70	413.8	368.5	101.6	96.6

Table 9. Vectorization speedups (IBM 3090-400) as a function of key vector length, the number of background coefficients. "Scalar with ESSL" means that only the ESSL subroutines were vectorized.

The remainder of the performance analysis concentrated on reducing the time spent within the SMINV routine. SMINV performs a single task, that of inverting a matrix, the sixe of which is directly dependent on the number of coefficients in the background fit.

As we have shown, the majority of the computational time is spent in the routine SMINV. This routine is part of a general matrix solution package, solving for example an equation such as Ax=B, where A and B are matrices and x is a vector. Within the ESSL library there are routines designed to perform just such a task. The next stage of vectorization was therefore to incorporate these into ROBFIT. The two

routines used were DGEF, a general matrix factorization code and DGES, a general matrix solver. A single call to DGEF followed by a call to DGES replaced the old matrix slover routines. The whole code was once again tested after incorporation of these ESSL matrix routines. Results from these runs also are shown in Table 9. Two runs were performed. One is simply with the ESSL routines vector compiled and the rest of the code scalar compiled. The second was with all routines vector compiled. Almost all of the speedup can be seen to come from the ESSL routines. There is a slight advantage in vectorizing only the matrix solving part of the code for vector lengths less than thirty. However we anticipate the general use of the code will require vector lenghts on the order of fifty or so. In this region the best performance is brought about by vectorizing all routines.

6. Conclusions

ROBFIT is a powerful spectral analysis package which can be operated from any personal computer system linked to an IBM 3090 system. All code control and graphical viewing of results can be performed in a simple manner from the PC environment. Because of its original development on PC systems, the standard ROBFIT code is intrinsically efficient in operation. However, further speed enhancements can be achieved by running the code on the IBM 3090 VF system. Utilizing the Vector Facility of this machine offers the user an operational speedup of a factor of three or more for fitting that requires a relatively large number of constants in its background determination. The code therefore turns a personal computer (PC XT, AT, PS/2 or compatible) into a powerful analysis workstation with the minimum of effort.

References

1. R. L. Coldwell, "An iterative peak finding code," in Radiative properties of Hot Dense Matter, 315-349, Sarasota, FL (World Scientific, 1983) Davis, Hooper et al editors.

2. R. L. Coldwell, "Iterative codes for fitting complete spectra," Nucl. Inst. and Methods in Physics Research A242 (1986) 455-461

3. A. C. Rester, R. L. Coldwell, F. E. Dunnam, G. Eichhorn, J. I. Trombka, R. Starr and G. P. Lasche, "Gamma-Ray Observations on Supernova 1987A from Antarctica," Astrophysical Journal Letters (submitted), Table 1.

4. W. H. Press, B. P. Flannery, S. A. Teukolsy, W. T. Vetterling, Numerical Recipes, the Art of Scientific Computing, (Cambridge Univ. Press, 1986) pp 539-46

5. for example, see Dimitri Mihalas Stellar Atmospheres (Freeman, 1978) p. 279

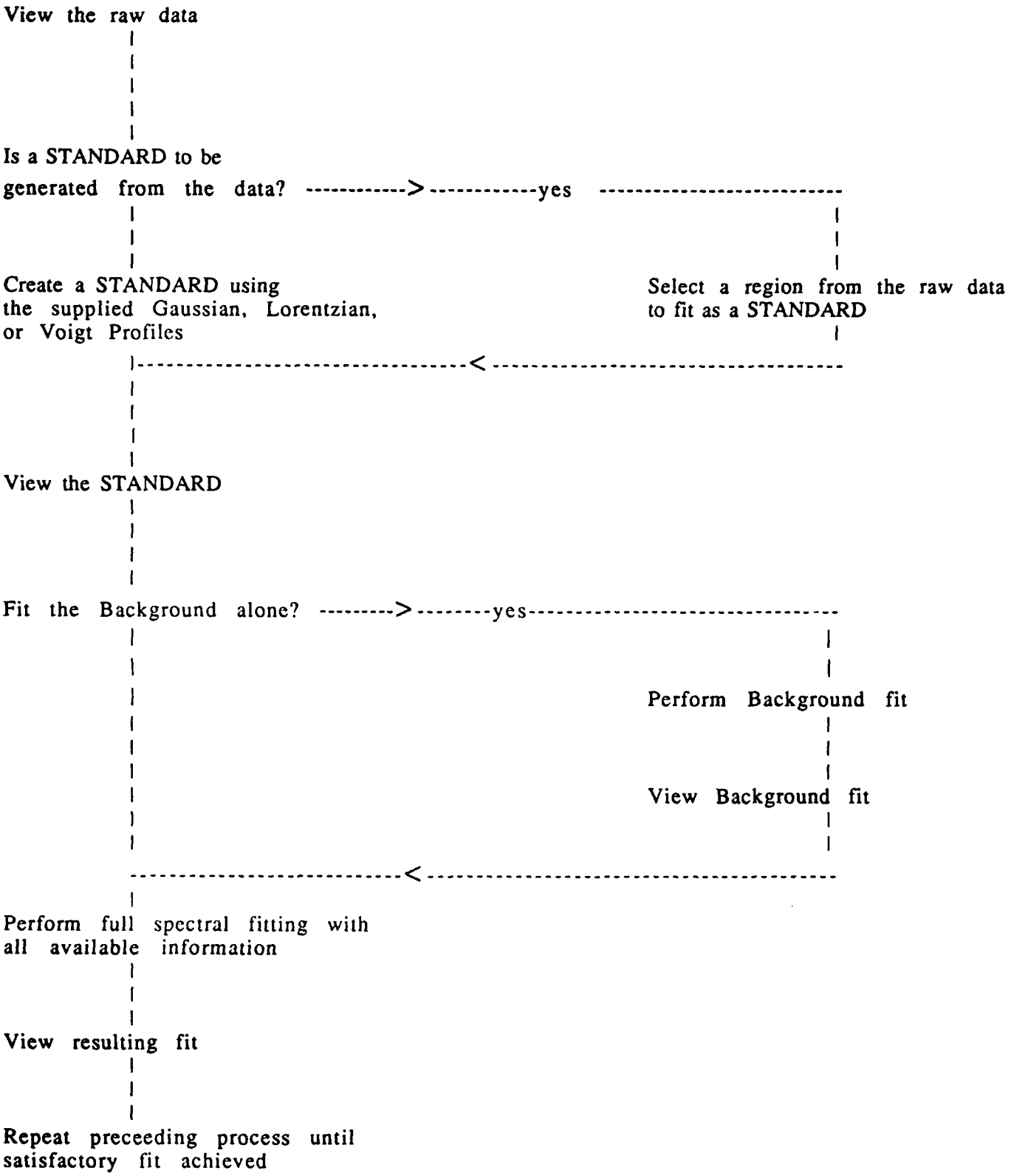


Figure 1. Overview of ROBFIT

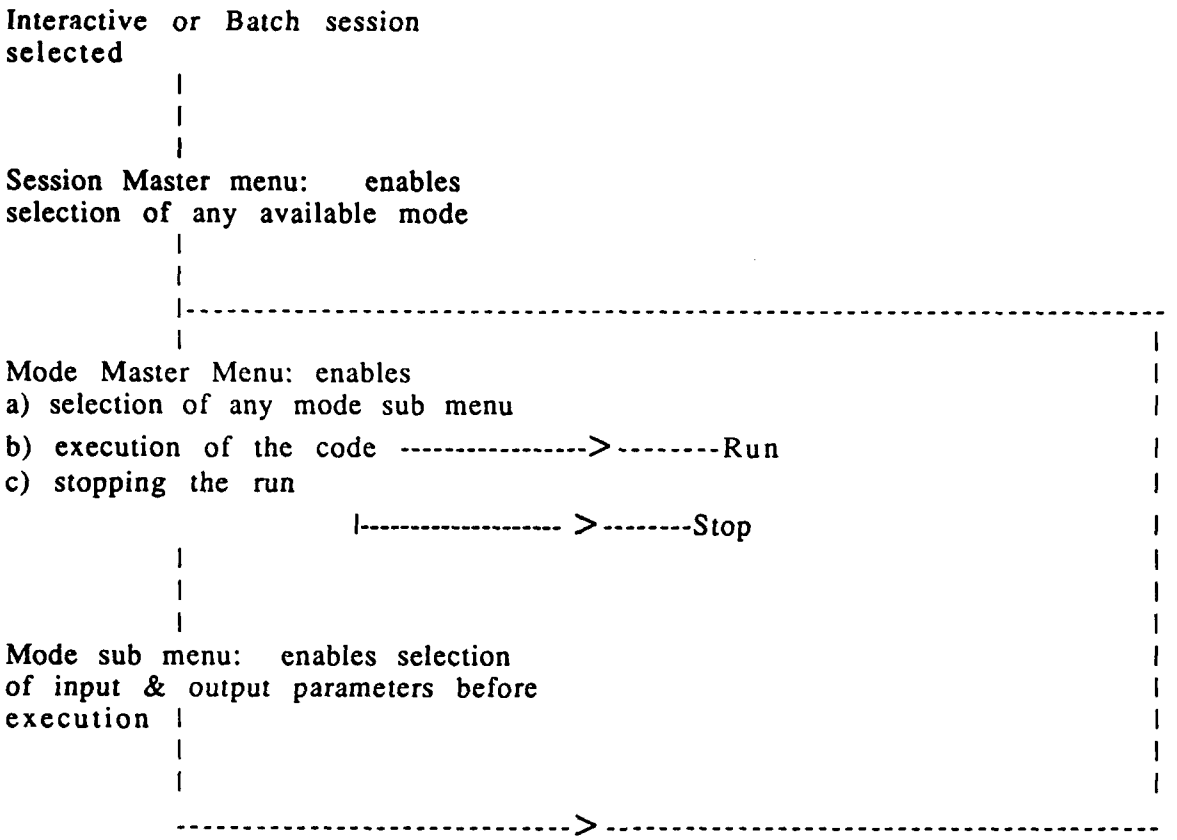


Figure 2. Overview of the ROBFIT menu driver.

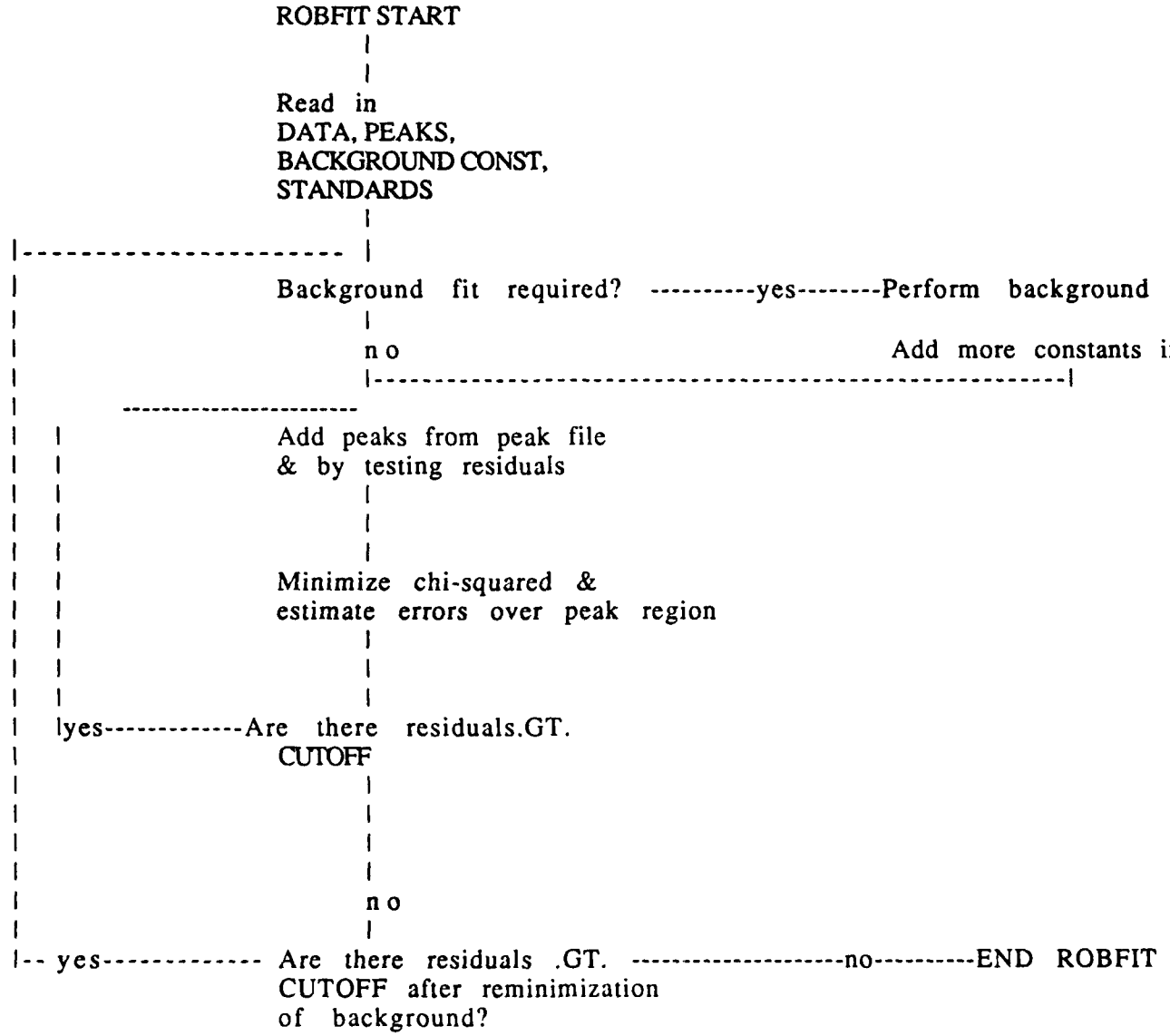


Figure 3. Overview of ROBFIT minimization procedure.

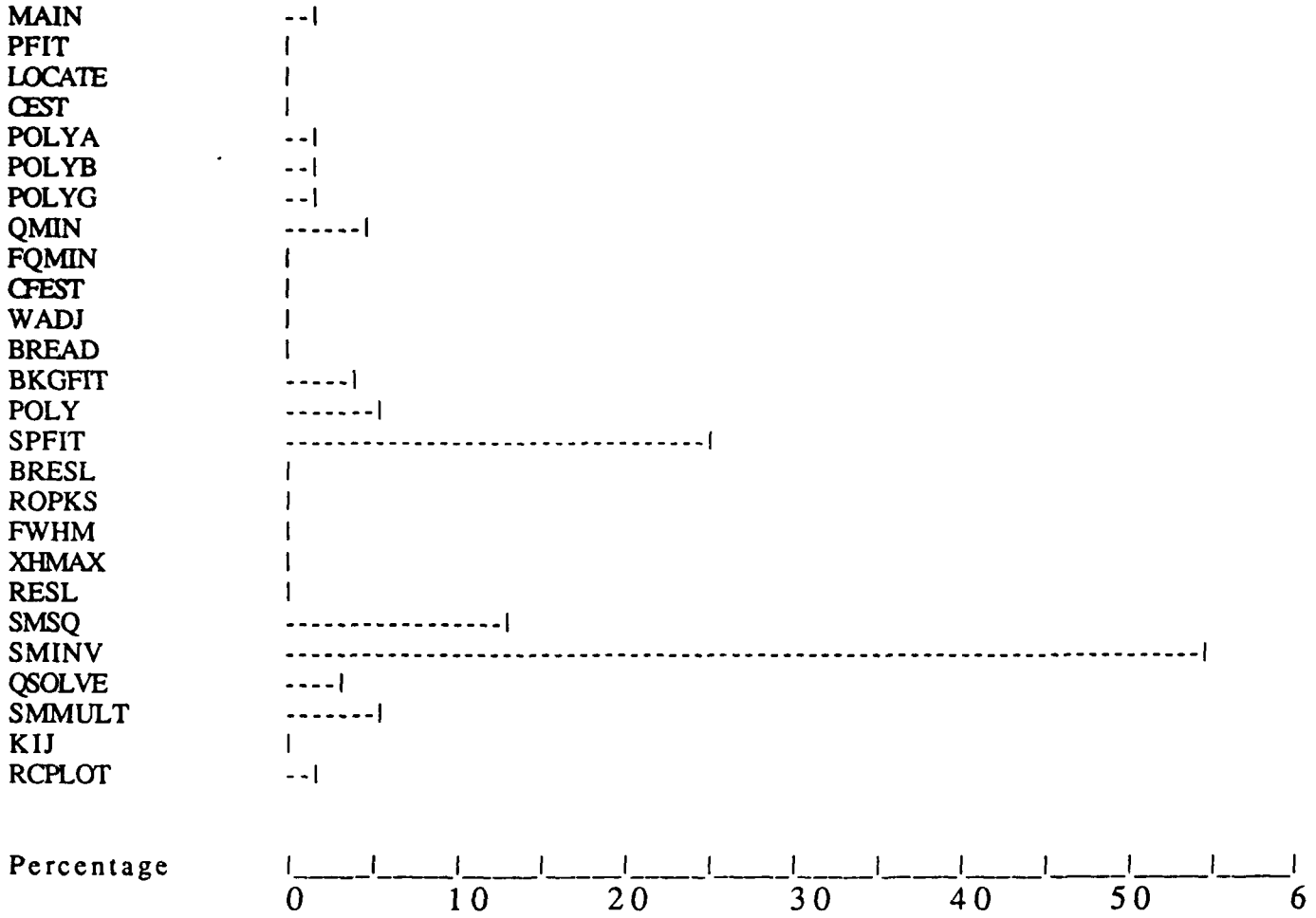


Figure 4. Percentages of time spent in eahc ROBFIT routine>

APPENDIX A. Menu page structure and description

Here we describe the layout of each of the mode menu pages that can be operated under ROBFIT. In order to reach the mode menus the user must have followed the startup sequence of ROBFIT. The first part of this sequence is to select either a batch or an interactive session, section 3 describes this selection process in more detail. After selecting a Session various mode options become available, the top-most master menu page allows the user to select the mode of operation from,

under CMS

1 Raw data display (RAWDD)

2 Standard generation (STGEN)

3 Standard display (STDIS)

4 Background fit (BKGFIT)

5 Full spectral fit (FSPFIT)

6 Full spectral display (FSPDIS)

or under MVS

1 Raw data display (RAWDD)

2 Background fit (BKGFIT)

3 Full spectral fit (FSPFIT)

4 Full spectral display (FSPDIS)

By entering the line number of the mode required, that particular mode can be selected for operation. Data fit display can only be performed under an interactive session, running displays under MVS produces a high quality laser printer output. Computer intensive tasks such as background and spectral fitting can be run either interactively or in batch. Operation of the MVS batch version of ROBFIT is outlined in more detail in Appendix C.

On selecting a mode the following menu structure is presented. The master menu for each mode is the first page to be displayed. From this point onward the user cannot re-select another mode of operation. If another mode is required ROBFIT must

be exited by entering 99 at the mode master menu level and re-running ROBFIT. The mode menus have the structure outlined in Figure 2.

From the mode master menu the user can select a sub-menu page related to this specific mode. The sub-menu pages contain the operating parameters for the mode, once in the sub-menu the user can alter any of these runtime parameters. Each sub-menu is accessed by entering the line number of that particular sub-menu.

At the sub-menu level there are four ways to proceed.

1) The first is to enter 0,0 this will exit the sub-menu and return the menu driver to the mode master menu, from which the mode can be run or exited.

2) The second is to delete a line, this is done by entering LINE NUMBER,1.

3) The third is to enter a new variable for one of the inputs, this is done by entering LINE NUMBER,10. The menu driver then asks for the new variable which must be entered on the next input line. After entry the driver will re-display the sub-menu with this change added.

4) The fourth and final option is to create a new line by entering LINE NUMBER (to create),100. This option is for use in adding in new standard information for a fit. Each standard must have its own specific set of *initialization data as outlined later.*

Description of the menu pages.

This section describes the menu pages for each of the modes of operation.

RAW DATA DISPLAY (RAWDD) MODE.

The RAWDD mode enables the user to view the raw data to be input into ROBFIT. This may be for diagnostic purposes or simply to select out a region of interest for analysis. The menu pages are as follows:

Page RAWDD

1 General information for the display.

2 Difference plot information.

3 Corrected plot information.

By selecting one of the three options the user can change the display requirements. Line one shows details on file input and display plotting options. Line two shows difference plot options. These pages are used if a comparison between two spectra is required. The difference plot subtracts one spectrum from another and plots the residuals. Line three is used to correct bad data. If data corruption has occurred for some reason and the user wishes to modify a certain region of the spectrum then line three shows the options for doing the modification.

Page RAWDD1 (General information)

1	Files to display f1,f2	Fname1 UF,Fname2 UF
2	Initial and final channels	1,default
3	Minimum and maximum peak heights	1,default
4	Log or linear scale	log
5	Title for the graph	Title

On line one the names of the two files to compare are entered. If there is just a single file then a backslash (\) is entered for the second filename. Line two contains the limits in channel numbers for the display. The default option displays all the available channels. On line three the range of peak heights is input. The default option takes these values from the data. On line four the user specifies whether a logarithmic or a linear Y scale is required. Line five contains the title that will be placed at the top of the display.

Page RAWDD 2 (Difference information)

1	Do you want a difference plot (y/n)	No
2	want to divide by the error (y/n)	No
3	the file name for sum	None
4	the file name for difference	None

5	min/max peak heights are	1,default
6	log or linear y scale	Log
7	normalization for the files	Default,default

On line one the user selects the difference option by specifying YES or NO. Line two gives the option of dividing by the error calculated from the difference between the files. This is for viewing purposes only; it helps reduce wide fluctuations that can occur when plotting the difference between two spectra. Line three is the file name in which the sum of the two data files is placed. This file is used in the full spectral fitting phase as the error on the difference data. Line four contains the file name for the difference data. Each file is again specified as FILENAME UF. Lines five and six specify the min/max peak heights and the log or linear choice for vertical scale respectively. The default for min/max is to the full range of the data. Line 7 contains the scaling parameter for the two input files. Channel values for the second file are scaled by (first scale value)/(second scale value).

Page RAWDD 3 (Correction information)

1	Do you want to correct the data (y/n)	No
2	channel range to be corrected.	1,default/1,default
3	file name for corrected data.	None/none
4	beginning and ending heights.	1,default/1,default

On line one the user selects the correction option by specifying YES or NO. Line two contains the channel range over which the correction is to be performed. Corrections can be made to both input files if needed. Line three contains the files for the output data. Both files must be specified as FILENAME UF. Line four contains the starting and ending channel heights for the selected region. The program will linearly interpolate over the troubled region between the selected heights on line four.

STANDARD GENERATION (STGEN) MODE.

The standard generation mode allows the use to create a standard peak from the raw data or to use one of the common peak shapes held within the program.

Page STGEN

1 General data on the generation

2 Selection of the fit type details

Line one shows the options for the standard shape selected. Line two shows the detailed data for the Voigt and data fit options.

Page STGEN1 (Generation data)

1	Name of the output standards file	Filename ST
2	Graphical output to be placed in file	SHAPE GR
3	Fit type (Gaus,Lore,Voig or Fitd)	Gaus
4	Number of background coeffs and splines	1,1

Line one contains the name of the file into which the standard output is to be placed. The main spectral fitting routine will use this file in its peak search phase. Line two contains the data for display of the standard. The filename in this case can not be changed by the user. If another name is required the user must rename the file after fitting. This information is used in the standard display mode of ROBFIT. On line three the user can specify the standard required. If either a Gaussian (Gaus) or a Lorentzian (Lore) is specified, then no further input is required. If Voigt (Voig) or a section of data (Fitd) is used to generate a standard then further options have to be set on page STGEN2. On line four the user selects the number of background coefficients and the number of splines to be used in the standard.

Page STGEN 2 (Type options)

1	Voigt fit ETA parameter	1
2	Data fit file name	Filename UF
3	and beginning/ending channel nos	1,default

Line one shows the Voigt ETA parameter that will be used in the generation. On line two the user specifies the file name in which the raw data resides and on line three the channel limits over which the standard is to be fitted. The default again being all the available range.

DISPLAY OF THE STANDARD (STDIS) MODE.

The standard display mode allows the user to view the standard peak generated with STGEN.

Page STDIS

1 Information for standard display.

2 Viewing information for the display.

Line one contains the general information for the display. Line two shows the options available during the display.

Page STDIS 1 (General information)

1	Name of the input graphics file	Filename GR
2	Beginning and ending channels	1,default
3	Desired min and max y scale	1,default
4	Log or linear y scale	Log

Line one contains the name of the file containing the input data for the plot. Line two shows the range of channels that will be displayed with the default being all channels. Line three shows the range over which the y scale will vary. The default option takes these values from the input data. Line four shows whether the plot will be linear or logarithmic in its y scale.

Page STDIS 2 (Viewing options)

1	View the curve fit (y/n)	Yes
2	View the splines (y/n)	Yes
3	View the background fit (y/n)	Yes

Answering yes to line one displays the curve fit. Similarly answering yes to line two shows the splines and line three shows the background under the standard.

BACKGROUND FIT (BKGFIT) MODE.

The background fit allows the user to do a fit to the background under the spectrum without the need to run the full spectral fitting code.

Page BKGFIT

1 Input data information.

2 Output data information.

Selection of the appropriate line shows the details for either input or output from the background fitter.

Page BKGFIT1 (Input details)

1	Maximum numberof background constants	100
2	Cutoff value set to	5
3	Input data file name Filename	UF
4	Beginning and ending channels for fit	1,default
5	Input background constant file	Filename CN,Yes
6	Weights 0=sqrt(data),1=calc,2=file	0
7	File containing the weights	None

Line one shows the maximum number of constants that will be used in the fit. Line two shows the cutoff at which the routine will stop fitting. Line three contains the raw data filename and line four the range over which the fit is to be performed. Again the default is all the available channels. Line five contains the input background constants file. The qualifier for this line selects either a logarithmic (YES) or a linear (NO) background fit. Line six selects the weights that are placed on the data points. A 0 takes the error to be the square root of the data, 1 determines the error form the spread of the data points over the entire selected region, 2 reads the errors in from an external file whose name is entered on line seven. The format of the file must be identical to the UF files as outlined in Appendix B.

Page BKGFIT2 (Output details)

1	Background constants output to	BKGCONS CN
2	Background fit	BKGCONS GR

Line one shows the file to which the constants for the background fit will be placed. Line two shows the file to which the background fit graphics information is sent.

FULL SPECTRAL FIT (FSPFIT) MODE.

This part of ROBFIT provides the means to fit the complete spectrum of peaks and background.

Page FSPFIT

1 General data on the spectral fit.

2 Input data information.

3 Output data information.

4 Standards files.

5 Peak type information.

Line one contains information which is most likely to be changed during a sequence of refits. Data such as the setting of the cutoff value and number of background constants can be found here. Line two holds the data which are to be input into the spectral fit. Similarly line three details where the output from the fit is to be placed. Line four shows a separate menu for standards files to be used with the fit while the line five menu contains the information on the width variations to be used for each of these standards.

Page FSPFIT1 (General fit information)

1	Maximum number of peaks	100
2	Starting,ending cutoffs, in N steps	2,1,2
3	VWID,FIXD and CONT,NONK,NOBF,FIXK	VWID,CONT
4	Maximum number of background constants	10

5	Weights 0=sqrt(data),1=calc,2=file	0
6	File containing the weights	None

Line one shows the maximum number of peaks that will be included in the fit the largest value being 255. Line two specifies the cutoff range and the number of steps that will be taken between the two limits. For example the values listed above will have a starting cutoff value of 2 and an ending cutoff of 1 and will take 2 steps between these two limits. Line three shows how the program will handle the peak width variation and background fitting. For example VWID selects a variable width fit while FIXD selects the fixed width format. CONT,NONK,NOBF and FIXK select continuous knot addition, no knot addition, no background fit and fixed knot option respectively. Line four shows the maximum number of background constants that will be used in the fit. Line five selects the error that is to be placed on the data. **0** selects an error that is the square root of the data point at each channel while **1** calculates the error from the spread of the data over the entire selected region. **2** allows the user to read in the errors from an external file whose name is specified on line six. This file must have the UF file format outlined in Appendix B.

Page FSPFIT 2 (Input data)

1	Input data file nameFilename	UF
2	Beginning and ending channels for fit	1,4000
3	Input peak file	Filename PK,YES
4	Input background constants file	Filename CN,YES

Line one shows the file name in which the raw data reside. Line two defines the channel range for the fit. The third line details the input peak file if there is one, NONE is specified if there is no peak file. If there is one the qualifier determines if the peak widths are accurate or not. For example, if there is no input peak file, as in the first cycle of fitting, then the user would specify NONE, NO. Line four details the input parameters for the background. The input file name is defined first, again, NONE specifies no input background file. The qualifier for Line four selects the type

of background fit. A logarithmic fit to the background is selected by specifying YES while a linear fit is selected by NO.

Page FSPFIT 3 (Output data)

1	Output file for background constants	Filename CN
2	Output file for Peak data	Filename PK
3	Output file for graph data	Filename GR

These three lines specify the files in which each of the fitted parameters are to be placed.

Page FSPFIT 4 (Standards data)

1	First standard	Filename ST
2	Second standard	Filename ST

This menu page details the standards to be used in the fit the files containing the standards are defined here.

Page FSPFIT 5 (Standards width variation)

1	Type 1 width,error and channel no.	5,1,100
2	Type 1 width,error and channel no.	10,1,1000
3	Type 2 width,error and channel no.	20,5,10
4	Type 2 width,error and channel no.	20,5,2000

etc.

Each pair of lines defines the limits over which a particular standard is to vary for example the first standard (type 1) has a peak width starting at 5+/-1 channels at around the 100 channel region and goes up to 10+/-1 channels at the 1000 channel region. Similarly the second standard has its variation specified in lines three and four. Each new standard has its width variation set on this menu page.

FULL SPECTRAL FIT DISPLAY (FSPDIS) MODE.

These pages allow the full spectral fit to be displayed. The routine uses as its input the file specified on menu page FSPFIT3 line 3.

Page FSPDIS

1 Information on the spectral fit display.

2 Viewing information for the display.

The first line provides general information for the display while the second line shows the screen setup for the plotting routines.

Page FSPDIS 1 (General information)

1	Name of the graphical file to be read	Filename GR
2	Beginning, ending channels for display	1.4000
3	Desired min and max y scale	1,default
4	Log or linear y scale	LOG

Line one contains the name of the file which has the graphical output. Line two specifies the channel range to be displayed. The default is specified as 1,DEFAULT and runs over the entire range of channels. Line three sets the min and max y axis range the default being the min and max values found in the data. Line four shows if the display is to be logarithmic or linear in its y axis.

Page FSPDIS 2 (Viewing options)

1	View the residuals (y/n)	Yes
2	View the curve fit (y/n)	Yes
3	View the peaks (y/n)	Yes
4	View the background fit (y/n)	Yes

These lines set up the screen that will be displayed the options can be switched off be selecting No as the option.

APPENDIX B. Filenames, Data input/output

RAWDD

1. Input Data

RAWDD1

There are two input files to be specified in the raw data display mode. They are found on line one of the RAWDD1 sub-menu. Options for selecting either one or two files have been discussed in the main text and will not be repeated here. Both files have the same format and must be specified as,

'Filename' UF

in the sub-menu where 'Filename' is at the user's option. The UF qualifier to the filename indicates that the files are written unformatted. These files must have been created using the following format.

```
WRITE(8)NBDP,XOFF
WRITE(8)(DATA(J),J=1,NBDP)
```

Here the data are written to unit 8 which is defined as an unformatted file. NBDP is the number of data points and XOFF is an offset number of channels. As an example of the use of XOFF consider a case in which the first 10 channels were corrupted data. The XOFF would be set to 11 so the data fit now only consider the channels between 11 and NBDP.

2. Output Data

RAWDD 2

Here two files must be selected. One for the sum data, the other for the difference data. They are both specified as

'Filename' UF

RAWDD 3

Here the files containing the corrected data must be selected. They are both specified as,

Filename' UF

3. Input Data

STGEN 2

There is a single input to be defined in the standard generation code. If the user wishes to create a standard from the data then the file containing the raw data must be specified on line two of the STGEN2 sub-menu. The format of this file is identical to that of the RAWDD1 input files and the file is again specified in the sub-menu as.

'Filename' UF

4. Output Data

STGEN 1

Two files need to be specified here. The first is the file containing the standard information which is used in the spectral fitting mode. It is specified as

Filename' ST

in the sub-menu. We use the ST qualifier to label standard information. Standards files are structured as follows.

Spline number	Spline constants		
1	0.100E+01	0.496E-03	0.108E+01
2	0.663E-01	-0.852E+00	0.553E+00
3	0.665E-01	0.852E+00	0.549E+00

The second output of STGEN1 is the file containing the graphics information. This file is input into the STDIS mode to allow the generated standard to be viewed, the file is specified as in the sub-menu.

SHAPE GR

We specify graphics files by using a GR qualifier in the filetype. This filename can not be altered by the user. If another filename is required the user must rename the SHAPE GR file after fitting.

STDIS

5. Input Data

STDIS1

There is a single input into the standard display mode. It is specified in line one of the STDIS1 sub-menu. This file is the graphics output of STGEN1, and is specified as,

SHAPE GR

6. Input Data

BKGFIT1

There are three input files to be specified on the BKGFIT1 sub-menu page. The first is the file containing the raw data which is input on line three of the sub-menu and is specified as,

Filename' UF'

It has a structure as outlined in the RAWDD inputs.

The second input file contains background constants from a previous iteration of the code. In the first iteration no file can be input, however, subsequent cycles of the code use constants files that are specified as,

'Filename' CN

where CN denotes a constants file. The structure of these files is shown in the following section on BKGFIT2 outputs.

The third input file contains the error values for each of the channels within the spectrum. The file is specified as,

'Filename" UF

7. Output Data

BKGFIT 2

Here there are two files to be understood. At this stage these files have a fixed naming convention, so that the file names can not be changed by the user. Their structure is given in the following. BKGCONS CN is the file which contains the background constants for the fit. It has a structure as follows (where this file contains ten constants):

1
507.753
-0.961
0.608E-03
-0.127E-06
-0.122E-06
1575.717
0.611E-06
927.665
-0.602E-06
946.139

This file can be fed back into a further iteration of the background fitting if necessary thus providing a starting point for the new fit. After fitting, the file can be renamed under VM/CMS to any user specified filename. If, during fitting, the input and output constants files are the same, the code will read the old constants then overwrite the file with the new constants.

The other output file is the BKGCONS GR file, this contains the graphics information for displaying the background fit. The file can be fed into the FSPDIS display pages as described in the main text.

FSPFIT

8. Input Data

FSPFIT 1

The file specified here contains the error values for each of the channels within the spectrum. The file is specifed as,

'Filename' UF

FSPFIT 2

Here there are three input files to be specified. The first is the raw data file which is input on line one of the FSPFIT2 sub-menu. This file is identical in structure to the RAWDD1 input files and is specified before as

'Filename' UF

The second file contains the peak data. These parameters are unknown on the first iteration of the code, so no file can be input. However, after the first cycle a peak file will be generated. Subsequent fitting can therefore be speeded up by utilizing this information. Feeding this output file back into the fit provides FSPFIT

with a starting point for the next iteration. Peak files are specified as,

'Filename' PK

We use PK to indicate a peak file. The structure of these files is illustrated in the section on FSPFIT3 output files.

The third input file of FSPFIT2 has the background constants. These data are identical in structure to that of the 'Filename' CN files described in the BKGFIT input section. Again this information may be unknown on the first iteration of the code and so no file can be input. If, however, a BKGFIT or FSPFIT run has been performed, then a constants file will have been generated. This file can then be used as input into FSPFIT on the next fit. These files are again specified as

'Filename' CN

or the user designated format. The structure of these file has been shown in the BKGFIT2 output section of this Appendix.

FSPFIT 4

Here the number of input files depends on the number of standards required to fit the data. Each file will have the same structure and is defined as

'Filename' ST

These files are the standards that have been created with STGEN and their structure is outlined in the section on STGEN1 outputs.

9. Output Data

FSPFIT 3

Here three output files need to be specified, namely,

Line 1 of sub-menu	Filename' CN	Background constants
Line 2 of sub-menu	Filename' PK	Peak parameters
Line 3 of sub-menu	'Filename' GR	Graphics data

CN files have already been discussed in the background fitting section, the

constants files output from FSPFIT are identical in structure to those in BKGFIT2 output.

The output peak files 'Filename' PK contain information on the fitted peaks. Their structure is as follows,

3 PEAKS 1000 CHAN, CHIS=923 . NITB=4

CUTOFF= 1.00

1.0560 1.0000

26.539	1.626	4.958	1.010	107.	73.	1
62..582	1.352	4.950	1.015	118.	67.	1
93 .161	1.448	4.569	1.012	90.	59.	1

etc for every peak found.

These numbers correspond to; peak position error in position, width error in width, strength error in strength and a flag which identifies with which standard the peak has been fitted.

Peak files can be edited by hand to change any of the peak parameters, for example, the position of a particular peak may be known before fitting. In this case the information on the position can be input into the fitting together with any information on width and strength estimates.

Finally, the file containing the graphics output must be specified. This file is for input into FSPDIS where viewing of the fit is performed.

FSPDIS

10. Input Data

FSPDIS 1

There is a single file input into the full spectral fitting display mode. It is input on line one of the FSPDIS1 sub-menu and is the graphics output file of FSPFIT3. It is specified as

'Filename' GR

APPENDIX C. MVS Operation

Under MVS the user can operate the two main fitting modes BKGFIT and FSPFIT and the spectral display mode FSPDIS. All menu pages operate as described in the main body of this guide, but all files are now replaced by permanent datasets. All files must be changes within the menu from,

'Filename' UF/GR etc.

to

'Filename'.UF/GR etc.

Initialization

Before running the code there are a few preliminaries that need to be taken care of. The first the loading into a library the compiled ROBFIT code. Loading is performed by submitting the CROBFIT JOB.

Where the user must specify the name for the compiled library. Before compilation is undertaken the user must ensure that the call to the ERRSET routine in the ROBFIT fortran is not commented out. In an interactive session this error recovery is specified in the ROBFIT EXEC. However, for a batch run it must be included in the FORTRAN.

Catalogue and create datasets

All named datasets used during ROBFIT operation must be catalogued beforehand. We have supplied an example job, CATDS JOB which performs this task. The user must specify the dataset to be catalogued. Once the dataset names have been decided, the job can be submitted with the command 'SUBMIT CATDS'.

For example, running the BKGFIT mode under MVS requires up to five datasets to be catalogued. These are

Filename.UF	(the raw data)
Filename.CN	(the input background constants)
BKGCONS.CN	(the output background constants)
F.UF	(copy of raw data)
FA.UF	(the fitted background)

These datasets need to be catalogued only once, unless they are deleted. Because BKGCONS. F and FA are used in every run of BKGFIT cataloguing them means that future runs of BKGFIT only require the data file and input background constants to be catalogued. A sample CATDS JOB is created each time a ROBFIT batch session is initiated. The user must therefore keep track of all created data sets and their status.

If the file format filename' SP is being used then these datasets must be catalogued with a format,

ALLOCATE (this line requires only a change to the dataset name).

SPACE=(TRK,(10,10)),BLKSIZE=6400,LRECL=80,RECFM=FB

CATLG DSNAME (again only a change to the datset name is required).

Other than this all operation is identical to the .UF file specification.

Uncatalogue and delete datasets

We also supply an example job that shows how to uncatalogue and delete any unwanted datasets. This job is called UNCATDS JOB. Again the user must specify the individual dataset to be deleted.

Listing all catalogued datasets

We have provided a job called CATLIST JOB which allows the user to list all the available datasets under a specific access and sequence number. To run the job the user must simply provide the access and sequence number by editing the JOB file.

Copying CMS files to MVS datasets

At the outset of a series of runs, files such as the raw data and any previously fitted parameters will need to be transferred onto the MVS machine. CMTOMV JOB is an example of how to perform the copy from CMS file to MVS dataset.

Copying MVS datasets to CMS files

Once ROBFIT has been run, the output files can be viewed by copying the MVS dataset into a CMS file. MVTOCN JOB is an example of how to perform this copy.

Running ROBFIT under MVS

Once all datasets have been allocated and the data copied to them ROBFIT can be run. MVS operation of the code is initiated by answering **Batch** to the startup sequence of the ROBFIT EXEC. The master menu then shows the four modes available under MVS. All mode menu pages operate as under CMS, but filenames now are replaced by dataset names. After all menu pages have been set up, the ROBFIT exec creates a file ROBFIT JOB. Submitting this file will load the selected fitting parameters and submit the job to the batch machine.

The background and spectral fitters perform the same operation as under VM/CMS. Running the spectral display mode here submits the graphical output to the 3820 laser printer.

APPENDIX D. User test cases

The test cases which follow have been selected to familiarize the user with the operation of ROBFIT. Each case has been selected from demonstration runs used to monitor the accuracy of the ROBFIT algorithms. All cases have been discussed in section 4 of this guide.

D1 Single peak fitting

Here a single gaussian peak has been generated and placed on an exponentially decaying background, as described in section 4.1.

The raw data reside in the ZTCASE1 UF file and can be viewed using the RAWDD mode.

Before running the fitting mode a standard must be generated using STGEN. We have supplied the gaussian standard used in the section 4 tests, ZGAUS3 ST, however, the user may wish to run the standard generator using the following setup;

Page STGEN1 (Generation data)

1	Name of the output standards file	TEST1 ST
2	Graphical output to be placed in file	TEST1 GR
3	Fit type (Gaus,Lore,Voig or Fitd)	Gaus
4	Number of background coeffs and splines	3,3

Page STGEN2 (Type options)

1	Voigt fit ETA parameter	1
2	Data fit file name	None
3	and beginning/ending channel nos	1,default

Once the standard has been created FSPFIT can be run, the FSPFIT mode menu pages must be set as follows;

Page FSPFIT1 (General fit information)

1	Maximum number of peaks.	100
2	Starting,ending cutoffs, in N steps	5,4,1
3	VWID,FIXD and CONT,NONK,NOBF,FIXK	VWID,CONT
4	Maximum number of background constants	4
5	Weights 0=SQRT(data),1=calc,2=file	0
6	File containing the weights	None

Page FSPFIT2 (Input data)

1	Input data file name.	ZTCASE1 SP
2	Beginning and ending channels for fit.	1,1000
3	Input peak file.	NONE, NO
4	Input background constants file.	NONE, YES

Page FSPFIT3 (Output data)

1	Output file for background constants	TCASE1 CN
2	Output file for Peak data	TCASE1 PK
3	Output file for graph data	TCASE1 GR

Page FSPFIT4 (Standards data)

1	First standard	ZTCASE1 ST

Page FSPFIT5 (Standards width variation)

1	Type 1 width, error and channel no.	50,10,10
2	Type 1 width,error and channel no.	50,10,900

On output the TCASE1 PK, TCASE1 CN AND TCASE1 GR files will contain all the fit data. TCASE1 PK should be identical to the ZTCASE1 PK file similarly TCASE1 CN will be identical to ZTCASE1 CN where the ZTCASE files are the outputs of our original runs.

The results of the curve fitting can be viewed by running FSPDIS with the following menu page setup;

Page FSPDIS1 (General information)

1	Name of the graphical file to be read	TCASE1 GR
2	Beginning,ending channels for display	1,1000
3	Desired min and max y scale	1,default
4	Log or linear y scale	LOG

Page FSPDIS2 (Viewing options)

1	View the residuals (y/n)	Yes
2	View the curve fit (y/n)	Yes
3	View the peaks (y/n)	Yes
4	View the background fit (y/n)	Yes

The display should produce a graph identical to that of ZTCASE1 GR.

D2 Minimum detectable peak.

i) Unambiguous peak determination in high noise region.

Here the ability of the fitting to find unambiguously, a small peak within the exponential background has been tested. The peak has been generated in the high noise region of the data as detailed in section 4.1.

Page FSPFIT1 (General fit information)

1	Maximum number of peaks.	100
2	Starting,ending cutoffs, in N steps	5,4,1
3	VWID,FIXD and CONT,NONK,NOBF,FIXK	VWID,CONT
4	Maximum number of background constants	4
5	Weights 0=SQRT(data),1=calc,2=file	0
6	File containing the weights	None

Page FSPFIT2 (Input data)

1	Input data file name.	ZTCASE2 SP
2	Beginning and ending channels for fit.	1,1000
3	Input peak file.	NONE,NO
4	Input background constants file.	NONE, YES

Page FSPFIT3 (Output data)

1	Output file for background constants	TCASE2 CN
2	Output file for Peak data	TCASE2 PK
3	Output file for graph data	TCASE2 GR

Page FSPFIT4 (Standards data)

1	First standard	ZTCASE1 ST

Page FSPFIT5 (Standards width variation)

1	Type 1 width,error and channel no.	50,10,10
2	Type 1 width,error and channel no.	50,10,900

On output the TCASE2 PK, TCASE2 CN AND TCASE2 GR files will contain all the fit data. TCASE2 PK should be identical to the ZTCASE2 PK file similarly TCASE2 CN will be identical to ZTCASE2 CN where the ZTCASE files are the outputs of our original runs.

The results of the curve fitting can be viewed by running FSPDIS with the following menu page setup;

Page FSPDIS1 (General information)

1	Name of the graphical file to be read	TCASE2 GR
2	Beginning,ending channels for display	1,1000
3	Desired min and max y scale	1,default
4	Log or linear y scale	LOG

Page FSPDIS2 (Viewing options)

1	View the residuals (y/n)	Yes
2	View the curve fit (y/n)	Yes
3	View the peaks (y/n)	Yes
4	View the background fit (y/n)	Yes

The display should produce a graph identical to that of ZTCASE 2 GR.

ii) Unambiguous peak determination in low noise region.

A similar test to i). above has been performed in the low noise region of the data, again the details are outlined in section 4.1. Fitting and display setups are;

Page FSPFIT1 (General fit information)

1 Maximum number of peaks.	100
2 Starting,ending cutoffs, in N steps	5,4,1
3 VWID, FIXD and CONT, NONK, NOBF, FIXK	VWID, CONT
4 Maximum number of background constants	4
5 Weights 0=SQRT(data),1=calc,2=file	0
6 File containing the weights	None

Page FSPFIT2 (Input data)

1 Input data file name.	ZTCASE2B SP
2 Beginning and ending channels for fit.	1,1000
3 Input peak file.	NONE, NO
4 Input background constants file.	NONE, YES

Page FSPFIT3 (Output data)

1 Output file for background constants	TCASE2B CN
2 Output file for Peak data	TCASE2B PK
3 Output file for graph data	TCASE2B GR

Page FSPFIT4 (Standards data)

1 First standard	ZTCASE1ST

Page FSPFIT5 (Standards width variation)

1	Type 1 width,error and channel no.	50,10,10
2	Type 1 width,error and channel no.	50,10,900

On output the TCASE2B PK, TCASE2B CN AND TCASE2B GR files will contain all the fit data. TCASE2B PK should be identical to the ZTCASE2B PK file. Similarly TCASE2B CN will be identical to ZTCASE2B CN where the ZTCASE files are the outputs of our original runs.

The results of the curve fitting can be viewed by running FSPDIS with the following menu page setup.

Page FSPDIS1 (General information)

1	Name of the graphical file to be read	TCASE2B GR
2	Beginning,ending channels for display	1,1000
3	Desired min and max y scale	1,default
4	Log or linear y scale	LOG

Page FSPDIS2 (Viewing options)

1	View the residuals (y/n)	Yes
2	View the curve fit (y/n)	Yes
3	View the peaks (y/n)	Yes
4	View the background fit (y/n)	Yes

The display should produce a graph identical to ZTCASE2B GR.

iii) Effect of peak width on detectability.

Here the width of the generated peak was made narrow to study its effect on detectability. Fitting and display setups are,

Page FSPFIT1 (General fit information)

1	Maximum number of peaks.	100
2	Starting,ending cutoffs, in N steps	5,4,1
3	VWID, FIXD and CONT, NONK, NOBF, FIXK	VWID, CONT
4	Maximum number of background constants	4
5	Weights 0=SQRT(data),1=calc,2=file	0
6	File containing the weights	None

Page FSPFIT2 (Input data)

1	Input data file name.	ZTCASE3 SP
2	Beginning and ending channels for fit.	1,1000
3	Input peak file.	NONE, NO
4	Input background constants file.	NONE, YES

Page FSPFIT3 (Output data)

1	Output file for background constants	TCASE3 CN
2	Output file for Peak data	TCASE3 PK
3	Output file for graph data	TCASE3 GR

Page FSPFIT4 (Standards data)

1	First standard	ZTCASE1 ST

Page FSPFIT5 (Standards width variation)

1	Type 1 width, error and channel no	5,3,10
2	Type 1 width, error and channel no.	5,3,900

On output the TCASE3 PK, TCASE3 CN AND TCASE3 GR files will contain all the fit

data. TCASE3 PK should be identical to the ZTCASE3 PK file. Similarly TCASE3 CN will be identical to ZTCASE3 CN where the ZTCASE files are the outputs of our original runs.

The results of the curve fitting can be viewed by running FSPDIS with the following menu page setup;

Page FSPDIS1 (General information)

1	Name of the graphical file to be read	TCASE3 GR
2	Beginning,ending channels for display	1,1000
3	Desired min and max y scale	1,default
4	Log or linear y scale	LOG

Page FSPDIS2 (Viewing options)

1	View the residuals (y/n)	Yes
2	View the curve fit (y/n)	Yes
3	View the peaks (y/n)	Yes
4	View the background fit (y/n)	Yes

The display should produce a graph identical to ZTCASE3 GR.

iv) Absolute limit of peak detection.

Here the cutoff value has been reduced to a level where spurious peaks are being picked up. At this level, however, ROBFIT can determine peaks which are burried well within the noise. Fitting and display setups are;

Page FSPFIT1 (General fit information)

1	Maximum number of peaks	100
2	Starting,ending cutoffs, in N steps	2,1,1
3	VWID,FIXD and CONT,NONK,NOBF,FIXK	VWID,CONT

4	Maximum number of background constants	4
5	Weights 0=SQRT(data),1=calc,2=file	0
6	File containing the weights	None

Page FSPFIT2 (Input data)

1	Input data file name.	ZTCASE3B SP
2	Beginning and ending channels for fit.	1,1000
3	Input peak file.	NONE, NO
4	Input background constants file.	NONE,YES

Page FSPFI3 (Output data)

1	Output file for background constants	TCASE3B CN
2	Output file for Peak data	TCASE3B PK
3	Output file for graph data	TCASE3B GR

Page FSPFIT4 (Standards data)

1 First standard ZTCASE1 ST

Page FSPFIT5 (Standards width variation)

1	Type 1 width,error and channel no.	5,3,10
2	Type 1 width,error and channel no.	5,3,900

On output the TCASE3B PK, TCASE3B CN AND TCASE3B GR files will contain all the fit data. TCASE3B PK should be identical to the ZTCASE3B PK file. Similarly TCASE3B CN will be identical to ZTCASE3B CN where the ZTCASE files are the outputs of our original runs.

The results of the curve fitting can be viewed by running FSPDIS with the

following menu page setup;

Page FSPDIS1 (General information)

1	Name of the graphical file to be read	TCASE3B GR
2	Beginning,ending channels for display	1,1000
3	Desired min and max y scale	1,default
4	Log or linear y scale	LOG

Page FSPDIS2 (Viewing options)

1	View the residuals (y/n)	Yes
2	View the curve fit (y/n)	Yes
3	View the peaks (y/n)	Yes
4	View the background fit (y/n)	Yes

The display should produce a graph identical to of ZTCASE3B GR.

D3 Multiple peak detection.

i) Unambiguous peak and separation determination.

Here the ability to determine unambiguously a peak at a given separation has been studied. These tests are outlined in section 4.2. Fitting and display setups are,

Page FSPFIT1 (General fit information)

1	Maximum number of peaks.	100
2	Starting,ending cutoffs, in N steps	5,4,2
3	VWID,FIXD and CONT,NONK,NOBF,FIXK	VWID,CONT
4	Maximum number of background constants	4
5	Weights 0=SQRT(data),1=calc,2=file	0

6	File containing the weights	None

Page FSPFIT2 (Input data)

1	Input data file name.	ZTCASE4 SP
2	Beginning and ending channels for fit.	1,1000
3	Input peak file.	NONE, NO
4	Input background constants file.	NONE, YES

Page FSPFIT3 (Output data)

1	Output file for background constants	TCASE4 CN
2	Output file for Peak data	TCASE4 PK
3	Output file for graph data	TCASE4 GR

Page FSPFIT4 (Standards data)

1	First standard	ZTCASE1 ST

Page FSPFIT5 (Standards width variation)

1	Type 1 width, error and channel no.	5,1,10
2	Type 1 width, error and channel no.	5,1,900

On output the TCASE4 PK, TCASE4 CN AND TCASE4 GR files will contain all the fit data. TCASE4 PK should be identical to the ZTCASE4 PK file similarly TCASE4 CN will be identical to ZTCASE4 CN where the ZTCASE files are the outputs of our original runs.

The results of the curve fitting can be viewed by running FSPDIS with the following menu page setup;

Page FSPDIS1 (General information)

1	Name of the graphical file to be read	TCASE4 GR
2	Beginning,ending channels for display	1,1000
3	Desired min and max y scale	1,default
4	Log or linear y scale	LOG

Page FSPDIS2 (Viewing options)

1	View the residuals (y/n)	Yes
2	View the curve fit (y/n)	Yes
3	View the peaks (y/n)	Yes
4	View the background fit (y/n)	Yes

The display should produce a graph identical to ZTCASE4 GR.

ii) Effect of cutoff value on separation determination.

Here we lower the cutoff value and detail its effects on detecting separated peaks. The fitting and display setups are;

Page FSPFIT1 (General fit information)

1	Maximum number of peaks.	100
2	Starting,ending cutoffs, in N steps	2,1.5,1
3	VWID,FIXD and CONT,NONK,NOBF,FIXK	VWID,CONT
4	Maximum number of background constants	4
5	Weights 0=SQRT(data),1=calc,2=file	0
6	File containing the weights	None

Page FSPFIT2 (Input data)

1	Input data file name.	ZTCASE5 SP
2	Beginning and ending channels for fit.	1,1000
3	Input peak file.	NONE, NO
4	Input background constants file.	NONE, YES

Page FSPFIT3 (Output data)

1	Output file for background constants	TCASE5 CN
2	Output file for Peak data	TCASE5 PK
3	Output file for graph data	TCASE5 GR

Page FSPFIT4 (Standards data)

1	irst standard	ZTCASE1 ST

Page FSPFIT5 (Standards width variation)

1	Type 1 width,error and channel no.	5,1,10
2	Type 1 width,error and channel no.	5,1,900

On output the TCASE5 PK, TCASE5 CN AND TCASE5 GR files will contain all the fit data. TCASE5 PK should be identical to the ZTCASE5 PK file. Similarly TCASE5 CN will be identical to ZTCASE5 CN where the ZTCASE files are the outputs of our original runs.

The results of the curve fitting can be viewed by running FSPDIS with the following menu page setup.

Page FSPDIS1 (General information)

1	Name of the graphical file to be read	TCASE5 GR
2	Beginning,ending channels for display	1,1000

3	Desired min and max y scale	1, default
4	Log or linear y scale	LOG

Page FSPDIS2 (Viewing options)

1	View the residuals (y/n)	Yes
2	View the curve fit (y/n)	Yes
3	View the peaks (y/n)	Yes
4	View the background fit (y/n)	Yes

The display should produce a graph identical to ZTCASE5 GR.

iii) Absolute minimum peak separation determination.

This test shows the limit to which peak separation determination can be driven. Fitting and display setups are as follows,

Page FSPFIT1 (General fit information)

1	Maximum number of peaks.	100
2	Starting,ending cutoffs, in N steps	2,1,2
3	VWID,FIXD and CONT,NONK,NOBF,FIXK	VWID,CONT
4	Maximum number of background constants	4
5	Weights 0=SQRT(data),1=calc,2=file	0
6	File containing the weights	None

Page FSPFIT2 (Input data)

1	Input data file name.	ZTCASE6 SP
2	Beginning and ending channels for fit.	1,1000
3	Input peak file	None,No
4	Input background constants file.	NONE, YES

Page FSPFIT3 (Output data)

1	Output file for background constants	TCASE6 CN
2	Output file for Peak data	TCASE6 PK
3	Output fjile for graph data	TCASE6 GR

Page FSPFIT4 (Standards data)

1	First standard	ZTCASE1 ST

Page FSPFIT5 (Standards width variation)

1	Type 1 width,error and channel no.	5,1,10
2	Type 1 width,error and channel no.	5,1,900

On output the TCASE6 PK, TCASE6 CN AND TCASE6 GR files will contain all the fit data. TCASE6 PK should be identical to the ZTCASE6 PK file. Similarly TCASE6 CN will be identical to ZTCASE6 CN where the ZTCASE files are the outputs of our original runs.

The results of the curve fitting can be viewed by running FSPDIS with the following menu page setup.

Page FSPDIS1 (General information)

1	Name of the graphical file to be read	TCASE6 GR
2	Beginning,ending channels for display	1,1000
3	Desired min and max y scale	1,default
4	Log or linear y scale	LOG

Page FSPDIS2 (Viewing options)

1	View the residuals (y/n)	Yes
2	View the curve fit (y/n)	Yes
3	View the peaks (y/n)	Yes
4	View the background fit (y/n)	Yes

The display should produce a graph identical to ZTCASE6 GR.

D5 Complex background fitting.

Here we show how the background fitter can perform on extremely complicated functions. We take as a data sample the Yttrium spectrum and fit it according to the procedure outlined in section 4.3. The background fitter is initially set up as follows;

Page BKGFIT1 (Input details)

1	Maximum number of background constants	10
2	Cutoff value set to	20
3	Input data file name	ZTCASE7 SP
4	Beginning and ending channels for fit	1,default
5	Input background constant file	None, Yes
6	Weights 0=SQRT(data),1=calc,2=file	0
7	File containing the weights	None

Page BKGFIT2 (Output details)

1	Background constants output to	BKGCONS CN
2	Background fit	BKGCONS GR

The output constants for the first iteration are to be compared with those in

ZY10 CN. Further cycles of the fitter can be performed by increasing the number of constants on line 1 of BKGFIT1 and feeding the old constants BKGCONS CN back into the fit at line 5 of BKGFIT1. Each of the constants files up to a total of 98 are provided with the code and can be used for comparison with the test case output. These constants files have the naming convention ZY'no of constants' CN. The test case output can be viewed by feeding the BKGCONS GR file into the FSPDIS display mode as described in section 3.6.

D6 Fitting a complete spectrum.

The following is a fit to data that was used to calibrate the GRAD experiment used on the Antarctic supernova expedition ref 3. The spectrum is characterized by large numbers of peaks of varying shape all sitting on a complex background. The user must be warned that fitting this spectrum will take a considerable amout of CPU time. To increase the fitting speed the constants file ZTCASE8 CN can be read in on line 4 of FSPFIT2 on the menu pages.

Page FSPFIT1 (General fit information)

1	Maximum number of peaks.	255
2	Starting,ending cutoffs, in N steps	40,10,4
3	VWID,FIXD and CONT,NONK,NOBF,FIXK	VWID,CONT
4	Maximum number of background constants	32
5	Weights 0=SQRT(data),1=calc,2=file	0
6	File containing the weights	NONE

Page FSPFIT2 (Input data)

1	Input data file name.	ZTCASE8 SP
2	Beginning and ending channels for fit.	50,4096
3	Input peak file.	NONE, NO
4	Input background constants file.	NONE, YES

Page FSPFIT3 (Output data)

1	Output file for background constants	TCASE8 CN
2	Output file for Peak data	TCASE8 PK
3	Output file for graph data	TCASE8 GR

Page FSPFIT4 (Standards data)

1	First standard	YT ST
2	Compton standard	COMP ST

Page FSPFIT5 (Standards width variation)

1	Yttrium start width,error and channel no.	2.07,0.05,800
2	Yttrium end width,error and channel no.	2.68,0.1,1900
3	Compton width,error and channel no.	80,20,50
4	Compton width,error and channel no.	80,20,4000

On output the TCASE8 PK, TCASE8 CN AND TCASE8 GR files will contain all the fit data. TCASE8 PK should be identical to the ZTCASE8 PK file. Similarly TCASE8 CN will be identical to ZTCASE8 CN where the ZTCASE files are the outputs of our original runs.

The results of the curve fitting can be viewed by running FSPDIS with the following menu page setup;

Page FSPDIS1 (General information)

1	Name of the graphical file to be read	TCASE8 GR
2	Beginning,ending channels for display	1,1000
3	Desired min and max y scale	1,default
4	Log or linear y scale	LOG

Page FSPDIS2 (Viewing options)

1	View the residuals (y/n)	Yes
2	View the curve fit (y/n)	Yes
3	View the peaks (y/n)	Yes
4	View the background fit (y/n)	Yes

The display should produce a graph identical to ZYCASE3 GR.

Appendix C

NLFIT code listing

```
      IMPLICIT REAL *8 (A-H,O-Z)
C *** THIS ROUTINE DOES A WEIGHTED NON-LINEAR LEAST SQUARES FIT TO
C *** USER SUPPLIED DATA IN L DIMENSIONS.  THE ROUTINE POLY MUST
C *** RETURN THE PARTIALS OF THE FITTING FUNCTIONS WITH RESPECT TO
C *** THE CONSTANTS BEING VARIED.
C *** DATA IS SUPPLIED IN FREE FORMAT AS X1,X2,X3,...,FXY,EP WHERE EP IS
C *** THE EXPECTED ERROR IN THE VALUE OF FXY.
      DIMENSION PPCC(5050),PPCCI(5050),P(100),PC(100),CONS(100)
C *** 5050  IS 100*101/2 THE SIZE NEEDED TO FIT 100 CONSTANTS
      DIMENSION X(100),SM(100),XS(100),XB(100),Y(100),YS(100),YB(100)
      CHARACTER*64 NA,NADAT,NACOM
      DATA XS,XB/100*1.D32,100*-1.D32/,YS,YB/100*1.D32,100*-1.D32/
      DATA CONS/100*0.D0/
      PRINT*,' ENTER THE NAME OF THE DIRECTION FILE'
      READ(*,'(A)')NA
      OPEN(13,FILE=NA,STATUS='OLD')
C *** ND IS THE DIMENSION OF X, NV IS THE NUMBER OF CONSTANTS BEING
FITTED
C *** CONS ARE THE INITIAL ESTIMATES OF EACH CONSTANT
C *** SM IS THE RELATIVE STIFFNESS OF EACH CONSTANT, 1 FOR LINEAR OR
NEAR LINEAR
C *** 10**6 FOR KNOT LIKE IN SPLINES
      READ(13,'(A)')NADAT
        IT=INDEX(NADAT,',')
      IF(IT.GT.0)NADAT=NADAT(1:IT-1)
      READ(13,*)ND
      READ(13,*)NV
      READ(13,'(A)')NACOM
      DO 5 I=1,NV
5     READ(13,*)CONS(I),SM(I)
      PRINT'(A/(2G20.12))',' CONS, SM',(CONS(I),SM(I),I=1,NV)
        PRINT'('' FIT IS BEING MADE TO DATA IN FILE''/1X,A)',NADAT
      CLOSE(13)
      FR=0
      CHB=1.E32
      CHL=1.E33
12    CONTINUE
      DO 13 I=1,NV
13     PC(I)=0.
      NVD=NV*(NV+1)/2
      DO 15 I=1,NVD
   15 PPCC(I)=0.
      CHI=0.D0
      OPEN(8,FILE=NADAT,STATUS='OLD')
```

```
      N=0
20    READ(8,*,END=40)(X(I),I=1,ND),FXY,EP
      N=N+1
      DO 22 I=1,ND
      XS(I)=DMIN1(XS(I),X(I))
22    XB(I)=DMAX1(XB(I),X(I))
      CALL POLY(X,P,NV,ND,CONS,FA)
      CHI=CHI+((FXY-FA)/EP)**2
      W=1./EP**2
      K=0
      DO 25 I=1,NV
25    PC(I)=PC(I)-2*(FXY-FA)*P(I)*W
      DO 30 I=1,NV
      DO 30 J=1,I
      K=K+1
30    PPCC(K)=PPCC(K)+2*W*P(I)*P(J)
       GOTO 20
40    WRITE(*,101)NV,N,ND
      CLOSE(8)
101   FORMAT(' A FIT IS BEING ATTEMPTED USING',I5,' CONSTANTS'/' TO',I5,
     C' OR FEWER DATA POINTS IN',I5,' DIMENSIONS')
       IF(DABS((CHI-CHL)/CHL).LT.1.D-7) GOTO 50
      CALL SMSQ(CHI,CHB,CHL,PC,PPCC,PPCCI,FR,CONS,SM,NV,1)
      WRITE(*,1332)CHI,CHB,CHL,FR
1332  FORMAT(' CHI=',E13.6,' CHB=',E13.6,' CHL=',E13.6,' FR=',E13.6)
      WRITE(*,1235)(CONS(I),I=1,NV)
1235  FORMAT(' CONS'/(4G16.7))
      GOTO 12
50    K=0
      CALL GSOLVE(PPCC,PPCCI,PC,P,NV)
      OPEN(13,FILE=NA,STATUS='OLD')
      DO 55 I=1,3
55    READ(13,'(A)')
      NACOM=' CONS     RELATIVE NONLINEARITY    ERROR IN CONS'
      WRITE(13,'(A)')NA
      PRINT*,' CHI AT END OF FIT',CHI
      CHIT=CHI
      CHI=DMAX1(CHI,1.D0*(N-NV))
      DO 18 I=1,NV
      IYB=10
      IF(ND.LT.2)IYB=1
      K=K+I
      ER=DSQRT(PPCCI(K)*CHI/(N-NV))
      WRITE(13,'(2G20.12,3X,''+-'',G10.3)')CONS(I),SM(I),EP
18    PRINT*,CONS(I),' +-',ER
      WRITE(13,'('' CHI = '',G20.6)')CHIT
      READ(*,*)ITEST
      OPEN(12,FILE='FITMERR.DAT')
      OPEN(10,FILE='FIT.DAT')
      OPEN(11,FILE='FITPERR.DAT')
      DO 80 IY=1,IYB
      DELY=XB(2)-XS(2)/9
      X(2)=XS(2)+(IY-1)*DELY
      DELX=(XB(1)-XS(1))/99
      DO 80  I=1,99
      X(1)=XS(1)+(I-1)*DELX
      CALL POLY(X,P,NV,ND,CONS,FA)
      K=0
```

```
      DX=0
      DO 27 J=1,NV
      K=K+J
27    DX=DX-PPCCI(K)*P(J)*P(J)
      K=0
      DO 67 J=1,NV
      DO 57 L=1,J
      K=K+1
57    DX=DX+2*PPCCI(K)*P(L)*P(J)
67    CONTINUE
      FMDX=0
      FPDX=0
      DX=DSQRT(DX*CHI/(N-NV))
      FMDX=FA-DX
      FPDX=FA+DX
      WRITE(10,*) X(1),FA
      WRITE(11,*) X(1),FPDX
      WRITE(12,*) X(1),FMDX
80    CONTINUE
      STOP
      END
      INCLUDE POLY.RKSPLINE
      SUBROUTINE SMSQ(CHI,CHB,CHL,PC,PPCC,A,FR,CONS,SM,NT,NW)
        IMPLICIT REAL*8 (A-H,O-Z)
C *** PPCC AND A MUST BE REAL*8.  THE OTHER ARGUMENTS CAN BE
C *** DECLARED REAL*4 AND THE ROUTINE WILL STILL WORK
C THE ROUTINE FINDS THE NECESSARY CHANGES IN CONS FOR THE NEW
C VALUE OF CHI TO BE EQUAL TO FR*CHI
C CHI IS THE CURRENT VALUE OF CHI (THE QUANTITIY BEING MINIMIZED)
C CHB IS THE VALUE PREDICTED FOR CHI ON THE LAST CALL TO SMSQ
C CHL IS THE LAST VALUE OF CHI, SM IS A VECTOR CONTAINING THE
C RELATIVE SMOOTHING FOR EACH CONSTANT IN THE VECTOR CONS.
C PC IS THE SET OF FIRST DERIVATIVES OF CHI WITH RESPECT TO THE CONSTANT
C PPCC IS THE SET OF SECOND DERIVS (PPCC(I+J*(J-1)/2) FOR I < J)
C STORED IN PACKED FORM, SEE SMINV
C NOTE THAT WHEN CHB=CHL, THE ROUTINE CAN GET NO LOWER
C NT IS THE NUMBER OF CONSTANTS.  PPCC AND AM ARE UPPER TRIANGLE
C  NW DECIDES WHAT TO WRITE
        SAVE NTO,SPC,SPPCC,SCONS,AS
      DIMENSION CONS(1),SM(1),PC(1),PPCC(1),A(1)
     # ,BB(107),B(107),SPC(107),SPPCC(5778),SCONS(107),PPCCD(107)
C *** THE ABOVE MUST BE DIMENSIONED NT OR LARGER
      DATA X0,X1,X2,AS/3*0.D0,-1.E0/
      DATA NTO/0/,TOL/1.D-6/
        NDT=NT*(NT+1)/2
      IFL=0
      IRT=0
      IAB=0
      QB=1.D0
      ILEFT=0
      ICYCLE=0
       IAA=0
        IF(NT.GT.107)GO TO 2500
      IF(NT.NE.NTO)THEN
      NTO=NT
        CHB=1.D33
        CHL=1.D34
        ENDIF
```

```
      AS=DMAX1(DMIN1(AS+2,75D0),-37D0)
C SAVE THE LAST SET OF COEFFS IF CHI<CHL
      IF(NW.GE.1)WRITE(*,'(A,3G20.7)')' CHI,CHB,CHL',CHI,CHB,CHL
      IF(CHI.GT.CHL)GO TO 10
      IF(1.000001*CHI.LT.CHL)IR=0
        DO 8 I=1,NT
        BB(I)=CONS(I)
        SCONS(I)=CONS(I)
8       SPC(I)=PC(I)
        DO 9 I=1,NDT
9       SPPCC(I)=PPCC(I)
      IF((CHI-CHB)/(CHL-CHI+1.E-6).GT..1)FR=.1*(9*FR+1)
      IF((CHI-CHB)/(CHL-CHI+1.E-6).GT..5)GOTO 20
      FR=DMAX1(1.D-2,FR*FR)
      GOTO 20
10    CONTINUE
C RESET TO THE BEST COEFFS (BEWARE THE ROUTINE MUST HAVE RUN BEFORE)
      AS=DMIN1(AS+17.D0,75D0)
      FR=.25*(3.+CHB/CHL)
       IR=IR+1
       CHI=CHL
         DO 18 I=1,NT
        CONS(I)=SCONS(I)
        BB(I)=CONS(I)
18      PC(I)=SPC(I)
        DO 19 I=1,NDT
19      PPCC(I)=SPPCC(I)
      IF(NW.GE.1)WRITE(*,191)IR
191   FORMAT(' IR=',I5)
      IF(IR.GT.20)THEN
        CHB=CHI
        RETURN
        ENDIF
20    CONTINUE
      IF(NW.GE.2)WRITE(*,173)FR
173   FORMAT(' FR=',E10.3)
      CHB=CHI
25    DE=DEXP(AS)
        KI=0
      DO 32 I=1,NT
        KI=KI+I
      PPCCD(I)=PPCC(KI)
32    PPCC(KI)=PPCC(KI)+SM(I)*DE
38      CONTINUE
        CALL GSOLVE(PPCC,A,PC,B,NT)
        KI=0
        DO 40 I=1,NT
        KI=KI+I
40      PPCC(KI)=PPCCD(I)
      ICYCLE=ICYCLE+1
C *** NOTE THAT B HERE IS THE SAME AS X IN GSOLVE
52    AJP=CHI
      IF(NW.GE.4)WRITE(*,10017)(B(I),I=1,NT)
        KI=0
      DO 60 K=1,NT
        KI=KI+K
      IF(DABS(B(K)).GT.1.D10)B(K)=DSIGN(1.D10,B(K))
      AJP=AJP+B(K)*(PC(K)+0.5*PPCC(KI)*B(K))
```

```
      IF(K.EQ.1)GOTO 60
      KM=K-1
        KT=KI-K
      DO 55 M=1,KM
        KT=KT+1
55    AJP=AJP+PPCC(KT)*B(M)*B(K)
60    CONTINUE
      IF(NW.GE.3)WRITE(*,104)AJP
104   FORMAT(' AJP=',E20.12)
C ****** AJP WAS CALCULATED ABOVE, WILL BE TESTED BELOW *************
C **** IF WE CAN FIND AJP < FR*CHI, WE CAN INTERPOLATE IN AS TO
C **** FIND A VALUE FOR WHICH AJP=FR*CHI, IFL = 1 INDICATES THIS CONDITI
      DT=AJP/DMAX1(1.D-35,CHI)
      IF(NW.GE.3)WRITE(*,105)DT
105   FORMAT(' DT=',E20.12)
      IF(DT.GE.FR)IAB=1
      IF(DT.LT.FR)IFL=1
      IF(DT-1.E0.LT.1.E-7)GOTO 1000
      IF(NW.GE.1)WRITE(*,109)AJP,AS
109   FORMAT(' THE MATRIX INVERSION APPEARS WRONG, AJP,AS',2E15.6)
C *********** THIS INVERSION IS WRONG, BUT A PREVIOUS ONE WAS RIGHT THU
        IF(NW.GE.3)WRITE(*,3421)IAA,AS
3421    FORMAT(' IAA,AS',I5,E20.6)
      IF(IAA.EQ.1)GOTO 1000
C ********** THIS INVERSION IS WRONG AND WE HAVE NEVER HAD A CORRECT ONE
      IF(AS.GE.76.)THEN
        IR=I+1
        GOTO 10
        ENDIF
115   AS=AS+3.
      GOTO 25
120   CONTINUE
C FINDING X'S ON EACH SIDE OF FR*CHI
      IF(ICYCLE.GT.50)AS=DMIN1(75.D0,AS+30.E0)
      IF(ICYCLE.GT.30.OR.DABS((AJP-FR*CHI)/CHI).LT.TOL)GOTO 3117
      IF(AJP.GT.FR*CHI)GOTO 130
      IRT=IRT+1
       ILEFT=0
      AR=AS
      QR=AJP/CHI-FR
      GOTO 140
130   AL=AS
      QL=AJP/CHI-FR
      IRT=0
      ILEFT=ILEFT+1
140   X0=X1
      X1=X2
      IF(IFL*IAB.EQ.0)GOTO 1150
      IF(QB.GT.1.-FR)GOTO 115
      IF(DABS((AR-AL)).LT..1E-5)GOTO 3117
      AS=AL-QL*(AR-AL)/(QR-QL)
      X2=AS
      IF(NW.GE.2)WRITE(*,102)AL,QL,AR,QR,AS
102   FORMAT(' AL,QL',2E15.6,' AR,QR',2E15.6,' AS=',E15.6)
      IF(IRT LT.2.AND.ILEFT.LT.2)GOTO 25
C AITKENS EXTRAPOLATION
      ALPHA=(X1**2-X0*X2)*(2.*X1-X2-X0)/((2.*X1-X2-X0)**2+1.E-37)
      IF(ALPHA.LT.AL.AND.AL.LT.AR)ALPHA=.5*(AL+AR)
```

```
      IF(ALPHA.GT.AL.AND.AL.GT.AR)ALPHA=.5*(AL+AR)
      IF(ALPHA.LT.AR.AND.AR.LT.AL)ALPHA=.5*(AL+AR)
      IF(ALPHA.GT.AR.AND.AR.GT.AL)ALPHA=.5*(AL+AR)
      IF(DABS((ALPHA-AR)/(AR-AL)).LT..1E-1.OR.DABS((ALPHA-AL)/(AR-AL))
     # .LT..1E-1)ALPHA=.5*(AL+AR)
      AS=ALPHA
      IF(NW.GE.2)WRITE(*,103)AS
103   FORMAT(' AITKENS EXTRAPOLATION, AS',E15.6)
      ILEFT=0
      IRT=0
      GOTO 25
C ********* WE HAVE FOUND 0 < AJP < CHI ***********************
C ********* AND WE WANT TO SAVE THE BEST SET OF B'S **********
1000  QC=DABS(AJP/CHI-FR)
      IF(IAA.EQ.0)GOTO 1050
      IF(QC.GT.QB)GOTO 1120
1050  CHB=AJP
      IAA=1
      QB=QC
        ASB=AS
      IF(NW.GE.2)WRITE(*,117)QB,AJP,AS
117   FORMAT(' QB,AJP,AS',3E15.6)
      DO 1115 I=1,NT
1115  BB(I)=B(I)
      IF(QB.LT.TOL)GOTO 3117
1120  GOTO 120
C ********* TRY AGAIN WITH MORE OR LESS SMOOTHING **********
1150  AS=AS-4.0
        IF(IR.GT.1)AS=.5*(AS+ASB)
      IF(IAB.EQ.0)AS=AS+8.
      IF(AS.LE.-75.)GOTO 3117
      IF(AS.GE.76.)GOTO 3117
      GOTO 25
C ****************** EXITING WITH THE BEST COEFF'S FOUND ***********
3117  CONTINUE
      CHL=CHI
      DO 3119 I=1,NT
3119  CONS(I)=CONS(I)+BB(I)
      IF(NW.GE.3)WRITE(*,10017)(BB(I),I=1,NT)
10017 FORMAT(1X,6E13.6)
      IF(NW.GE.3)WRITE(*,10018)QB
10018 FORMAT(' THE FINAL QB IS',E20.13)
        IF(NW.GE.1)PRINT'(A,G11.4,' FINAL AS IS',AS
      RETURN
2500    WRITE(*,4367)NT
4367    FORMAT(' NT OF',I5,' IS TOO LARGE FOR THE'/
     # ' DIMENSION OF B AND BB AND ALSO TERMS IN GSOLVE')
        STOP
      END
        SUBROUTINE SMINV(AP,N,IFL)
        IMPLICIT REAL*8 (A-H,O-Z)
        DIMENSION AP(1)
C *** THIS ROUTINE BEGINS IN LINPACK CHAPTER 3 - POSITIVE
C *** DEFINITE SYMMETRIC MATRICES
C *** IN OUR CASE (CURVE FITTING) WE CAN ALWAYS WRITE THE APPROXIMATE
C *** FUNCTION AS SUM C*SJ(XI).  IF WE THEN FORM X**2 = SUM ON I
C *** OF SUM ON J (CJ*SJ(XI))**2 WE FIND X**2 = SUM ON K CK *
C *** SUM ON J CJ * AJK WHERE AJK IS SUM ON I SJ(XI)*SK(XJ), THE
```

```
C *** POSITIVE DEFINITE SYMMETRIC MATRIX OF INTEREST HERE
C
C *** AP IS A PACKED MATRIX AS IN THE FOLLOWING
C *** K=0
C     DO 20 J=1,N
C     DO 20 I=1,J
C     K=K+1
C     AP(K)=A(I,J)
C 20  CONTINUE
        IFL=0
        JJ=0
        DO 30 J=1,N
        S=0
        KJ=JJ
        KK=0
        IF(J.LT.2)GOTO 20
        JM1=J-1
        DO 10 K=1,JM1
        KJ=KJ+1
        T=AP(KJ)
        KM1=K-1
        IF(KM1.LT.1)GOTO 8
        DO 5 I=1,KM1
5       T=T-AP(I+KK)*AP(I+JJ)
8       KK=KK+K
        T=T/AP(KK)
        AP(KJ)=T
        S=S+T*T
10      CONTINUE
20      CONTINUE
        JJ=JJ+J
        SMT=DMAX1(S,AP(JJ))
        DIFF=AP(JJ)-S
        IF(DIFF.GT.1.D-10*SMT.AND.DIFF.GT.1.D-70)GOTO 25
C        WRITE(*,1975)J,AP(JJ),S
1975    FORMAT(' POS DEF QUANT LE 0. J=',I5,' AP(JJ)=',E10.3,
     # ' S=',E10.3)
C *** FIX IS MORE SMOOTHING FOR A(J,J)
        IFL=-1
        DIFF=1.D30
25      AP(JJ)=DSQRT(DIFF)
30      CONTINUE
C *** NEXT WE CONSTRUCT THE INVERSE MATRIX
        KK=0
        DO 100 K=1,N
        K1=KK
        KK=KK+K
        AP(KK)=1/AP(KK)
        T=-AP(KK)
        KM=K-1
        IF(KM.LE.0)GOTO 86
        DO 85 I=1,KM
85      AP(I+K1)=T*AP(I+K1)
86      KP1=K+1
        J1=KK
        KJ=KK+K
        IF(N.LT.KP1)GOTO 90
        DO 80 J=KP1,N
```

```
          T=AP(KJ)
          AP(KJ)=0
          DO 70 I=1,K
70        AP(J1+I)=AP(J1+I)+T*AP(K1+I)
          J1=J1+J
          KJ=KJ+J
80        CONTINUE
90        CONTINUE
100       CONTINUE
          JJ=0
          DO 130 J=1,N
          J1=JJ
          JJ=JJ+J
          JM1=J-1
          K1=0
          KJ=J1+1
          IF(JM1.LT.1)GOTO 120
          DO 110 K=1,JM1
          T=AP(KJ)
          DO 105 I=1,K
105       AP(I+K1)=AP(I+K1)+T*AP(I+J1)
          K1=K1+K
          KJ=KJ+1
110       CONTINUE
120       CONTINUE
          T=AP(JJ)
          DO 125 I=1,J
125       AP(I+J1)=     T*AP(I+J1)
C ABOVE LINE ORIGINALLY HAD ANOV
130       CONTINUE
140       CONTINUE
          RETURN
          END
          FUNCTION ANOV(X,Y)
          IMPLICIT REAL*8 (A-H,O-Z)
          EQUIVALENCE (IX,XT),(IY,YT)
C         ANOV='FFFF7CFF'X*DSIGN(1.D0,X)*DSIGN(1.D0,Y)
          XT=X
          YT=Y
C         IF((IX.AND.'7F80'X)+(IY.AND.'7F80'X).LT.'BC80'X)
C     #   ANOV=XT*YT
           ANOV=XT*YT
          RETURN
          END
          SUBROUTINE GSOLVE(A,AM,B,X,NT)
          IMPLICIT REAL*8 (A-H,O-Z)
C *** SOLVES AX=-B IN CASES WHERE THE MATRIX CAN BE INVERTED CHEAPLY
C *** AND ACCURATELY WITH NO PROBLEMS
          DIMENSION A(1),AM(1),B(1),X(1)
          NTD=NT*(NT+1)/2
          DO 10 I=1,NTD
10        AM(I)=A(I)
          CALL SMINV(AM,NT,IFL)
          CALL SMMULT(AM,B,X,NT)
          DO 20 I=1,NT
20        X(I)=-X(I)
          RETURN
          END
```

```
      SUBROUTINE SMMULT(A,B,C,N)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION A(1),B(1),C(1)
C *** FOR PACKED MATRICES C(N)=A(N*(N+1)/2)*B(N)
      DO 10 I=1,N
10    C(I)=0
      K=0
      DO 20 I=1,N
      IM=I-1
      DO 15 J=1,IM
      K=K+1
      C(I)=C(I)+A(K)*B(J)
15    C(J)=C(J)+A(K)*B(I)
      K=K+1
20    C(I)=C(I)+A(K)*B(I)
      RETURN
      END
      FUNCTION KIJ(I1,I2)
C      WRITE(*,'(A,2I5)')' IN KIJ I1,I2',I1,I2
      I=MIN0(I1,I2)
      J=MAX0(I1,I2)
      KIJ=(J*(J-1)/2)+I
      RETURN
      END

C *** Then substitute one of the following poly routines as required.

C *** For Randomly placed splines use
      SUBROUTINE POLY(X,P,NV,ND,CONS,FA)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION P(1),X(1),CONS(1)
      DIMENSION AK(40)
      SAVE AK,NC
      DATA NC/0/
      NKN=NV-4
      IF(NC.GE.1)GOTO 15
      NC=1
      PRINT*,' ENTER ',NKN,' KNOT LOCATIONS'
      READ(*,*)(AK(J),J=1,NKN)
15    P(1)=1
       DO 20 I=2,4
20     P(I)=X(1)*P(I-1)
      DO 30 I=5,NV
      XM=AK(I-4)-X(1)
      P(I)=0
      IF(XM.LT.0.D0)GOTO 30
      P(I)=XM**3
30    CONTINUE
      FA=0
      DO 32 I=1,NV
32    FA=FA+CONS(I)*P(I)
      RETURN
      END

C *** For fixed knot splines use

      SUBROUTINE POLY(X,P,NV,ND,CONS,FA)
      IMPLICIT REAL*8 (A-H,O-Z)
```

```
      DIMENSION P(1),X(1),CONS(1)
      DIMENSION AK(40)
      NKN=NV-4
      IF(NKN.GT.0)AKN=40./NKN
      DO 10 I=1,NKN
10    AK(I)=I*AKN
       P(1)=1
       DO 20 I=2,4
20     P(I)=X(1)*P(I-1)
      DO 30 I=5,NV
      XM=AK(I-4)-X(1)
      P(I)=0
      IF(XM.LT.0.D0)GOTO 30
      P(I)=XM**3
30    CONTINUE
      FA=0
      DO 32 I=1,NV
32    FA=FA+CONS(I)*P(I)
      RETURN
      END

C *** For a spline fit use

       SUBROUTINE POLY(X,P,NV,ND,CONS,FA)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION P(1),X(1),CONS(1)
      DIMENSION AK(40)
      NKN=NV-4
       P(1)=1
       DO 20 I=2,4
20     P(I)=X(1)*P(I-1)
      DO 30 I=5,NV,2
      XM=CONS(I+1)-X(1)
      P(I)=0
      P(I+1)=0
      IF(XM.LT.0.D0)GOTO 30
      P(I)=XM**3
      P(I+1)=3*CONS(I)*XM*XM
30    CONTINUE
      FA=0
      DO 32 I=1,4
32    FA=FA+CONS(I)*P(I)
      DO 34 I=5,NV,2
34    FA=FA+CONS(I)*P(I)
      RETURN
      END

C *** For a polynomial fit

      SUBROUTINE POLY(X,P,NV,ND,CONS,FA)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION P(1),X(1),CONS(1)
       P(1)=1
       DO 10 I=2,NV
10     P(I)=X(1)*P(I-1)
      FA=0
      DO 32 I=1,NV
32    FA=FA+CONS(I)*P(I)
```

```
      RETURN
      END
```

Distribution List

Scientific Officer Code: 1114SP — 2 copies
R. Gracen Joiner
Office of Naval Research
800 North Quincy Street
Arlington, Virginia 22217-5000

Grant Administrator — 1 copy
Office of Naval Research
Resident Representative N66020
Georgia Institute of Technology
206 O'Keefe Building
Atlanta, GA 30332-0490

Director, Advanced Research Projects Agency — 2 copies
1400 Wilson Boulevard
Arlington, Virginia 22209
ATTN: Program Management

Director, Naval Research Laboratory — 1 copy
ATTN: Code 2627
Washington, DC 20375

Defense Technical Information Center — 1 copy
Building 5, Cameron Station
Alexandria, Virginia 22314